Low dimensionality spectral sensing for low cost material discrimination and identification

by

Andrew Matthew Bardagjy

B.S.E.E., Georgia Institute of Technology (2010)

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of

Masters of Science in Media Arts and Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2013

Author [illegible]
Program in Media Arts and Sciences
September 12, 2012

Certified by [illegible]
V. Michael Bove, Jr.
Principal Research Scientist
Media Lab
Thesis Supervisor

Accepted by [illegible]
Pattie Maes
Associate Academic Head
Program in Media Arts and Sciences

Low dimensionality spectral sensing for low cost material discrimination and identification

by

Andrew Matthew Bardagjy

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
on September 12, 2012, in partial fulfillment of the
requirements for the degree of
Masters of Science in Media Arts and Sciences

Abstract

Spectroscopy is a powerful tool in material identification, characterization and discrimination. Unfortunately industrial and laboratory spectrometers are typically very large, costly, and inconvenient. The aim of this thesis is to broaden the awareness and appeal of spectroscopic sensing modalities by exploring specialized, rather than general purpose instruments. Rather than sensing the entire spectrum, these devices work by observing just the particular spectral features needed to perform identification or discrimination. This approach greatly simplifies the instrument reducing the cost, size, power consumption, and analysis complexity by many orders of magnitude. In this work the anatomy of such specialized sensors is explored by way of a thorough discussion of illuminators, current sources, photodetectors, photodiode amplifiers, control systems and part selection. In the following chapters, instruments are designed and fabricated, and their tradeoffs are enumerated and discussed. Finally, these building-blocks are combined to construct several working prototypes which are informally characterized.

Thesis Supervisor: V. Michael Bove, Jr.
Title: Principal Research Scientist, Media Lab

Low dimensionality spectral sensing for low cost material discrimination and identification

by

Andrew Matthew Bardagjy

The following people served as readers for this thesis:

Thesis Reader [illegible]

Joseph A. Paradiso
Associate Professor of Media Arts and Sciences
Program in Media Arts and Sciences

Thesis Reader [illegible]

Andrew B. Lippman
Senior Research Scientist
Program in Media Arts and Sciences

Contents

List of Figures

Chapter 1

Introduction

Spectroscopy, the interaction of light and matter, is a powerful analytic tool in the study of material characteristics and material identification. In spectroscopy, measurements are typically shown as plots of intensity on a wavelength or frequency. Atoms, molecules, crystals, and nuclei interact uniquely with different photon energies or wavelengths. This property is exploited to detect, characterize, and identify materials. In addition to basic science, spectroscopy has been applied in fields as varied as agriculture, medicine, material science, defense and food science.

Traditionally, spectrometers are general purpose instruments that are designed to characterize some sample over a continuum of wavelengths. These spectrometers are impressive because of their wide applicability. For example, the same instrument used to characterize pigments can also be used to differentiate whiskey from scotch.

This approach, while capable of amazing insights, has a number of notable drawbacks. Because the instruments are general purpose, a number of considerations must be made to accommodate broadband measurements. In particular, these instruments produce a terrific amount of data, are rather large, and quite expensive. Despite heroic cost reduction and miniaturization efforts, the least expensive spectrometers still cost thousands of dollars and the smallest instruments are $100\,\text{cm}^2$ to $1000\,\text{cm}^2$. As such, spectroscopy remains today exclusively in the realm of scientific and industrial research.

The aim of this thesis is to broaden the application of spectroscopic sensors by designing specialized sensors for particular use cases. In particular, a framework is presented whereby a specialized sensor can be designed given a specific use case. Because these sensors are specialized rather than general purpose, they are smaller, less expensive, and their results are simpler to interpret. These sensors aspire to bring what was once an inaccessible scientific instrument to the average consumer, influencing fields as diverse as safety, medicine and food science.

1.1 Background

The history of spectroscopy begins with early experiments of Sir Isaac Newton. He demonstrated that the light from the sun was composed of a continuum of wavelengths. In those experiments, Newton employed glass prisms to disperse the light in a manner similar to the way water particles disperse light to form rainbows. These experiments, described in his book *Opticks*, also demonstrate that the refractive index of glass is frequency dependent, yielding wavelength dispersion [32].

In coming years, Joseph Fraunhofer observed that the sun's spectrum, when carefully dispersed, was crossed by a number of dark bands. He spent many years cataloging and characterizing these notches, assigning letters to the most prominent features [24].

These bands, now known as Fraunhofer Lines, indicate portions of the solar spectrum which are largely absorbed by the Earth's atmosphere. Decades later, Gustav Kirchhoff concluded that by examining the position and relative intensity of these lines, it is possible to ascertain the chemical composition of the atmosphere [29].

In the early 1800's, scientists including Herschel, Talbot, Wheatstone, and Angstrom discovered that one can identify materials by observing the spectrum of transmitted reflected or emitted light [8]. Simple, human observable examples of this phenomena are: metal typically appears a dull gray, wood is brown and healthy vegetation is green.

There are many examples of differing materials that appear similar to a human observer but radically different to a spectroscopic instrument. For example, is a red jacket made of nylon or cotton? Is the blue paint oil or water based? Is the table made of plastic or marble?

80
Joseph von Fraunhofer Optiker und Physiker 1787-1826 Deutsche Bundespost
1987
(a) Commemorative stamp celebrating the discoveries of Fraunhofer (*Wikimedia Commons*).

(b) Sodium carbonate flame test (*Wikimedia Commons*).

(c) A modern spectrometer (*Agilent*).

While these materials may be indistinguishable to a human's three coarse color receptors, they are readily differentiable when observing the finer details of their reflected or transmitted spectrum. Unfortunately, commercial spectrometers are typically large, complex and expensive restricting their use to laboratory operation. This is largely because these instruments are designed for general purpose characterization of arbitrary spectra.

A crucial observation is, for many of the desired use cases of reflectance or transmission spectra, *arbitrary* spectral measurement is not required. For example, the blood of a hypoglycemic and healthy patient can be easily differentiated by measuring only the spectral components corresponding to glucose [6].

This insight is exploited in this work to design spectral sensing devices, whereby only pertinent spectral bases are measured. Due to their specialization, these sensors promise to be smaller, lower-cost, and have simpler data inference.

1.2 Related Works

Classic techniques include reflection and transmission spectroscopy over wavelengths ranging from high-energy X-rays to long radio waves [8, 19]. Recently, with the invention of the laser, nonlinear spectroscopic techniques such as second-harmonic generation [14], stimulated Raman spectroscopy [21] and four-wave-mixing [42] have further increased the accuracy of spectroscopic measurements. These techniques, while advanced, typically strive to maximize accuracy rather than minimize cost - subsequently, they exist only in the realm of high end research laboratories.

(d) An admittedly large NMR spectrometer (*Wikimedia Commons*).

(e) X-Ray fluorescence spectrometer (*Thermo-Scientific*).

(f) Blood oxygenation sensor (*Jober International*).

Other, more esoteric methods have been devised to discern material composition. These include nuclear magnetic resonance (NMR) spectroscopy [35], ellipsometry and polarimetry [2], and mass spectroscopy [41]. These methods, while typically even more accurate than optical methods, are found even more exclusively in laboratory environments.

Some portable devices to have been made that perform materials identification. For example, handheld X-ray fluorescence spectrometers are frequently used in the field to identify metals [7]. These devices are mostly used for metals, as they have difficulty identifying the presence of lighter atoms. Further, they are unsuitable for repeated use on biological samples, as X-Rays can damage tissue.

In recent years, there has been a push to develop low cost, portable, visible light spectrometers [9]. Despite many impressive engineering accomplishments, these devices remain costly. Furthermore, rather than focusing on a particular use case, developers strive to create universal sensors which increases the complexity and, therefore, the cost of the device. These also require expert human or computer data analysis, making them inaccessible to the average consumer.

Indeed, there has been a substantial effort to analyze and visualize output from general purpose spectral sensors. The data are often used for material classification through both supervised [11] and unsupervised [12] learning. In general, these algorithms work to tease salient features from complete spectral measurements, while special purpose spectrometers sample those features directly.

Also, some researchers have explored differing modalities of multispectral data representation and presentation. Typically spectroscopy data is displayed as a false-color RGB image, though there has been pioneering work in 3D data visualization [16], and synesthetic representation [13] where data is mapped to other senses.

That said, there have been a few devices which have been designed to exploit spectral characteristics of materials for a specialized purpose. Most notable of those is the blood oxygenation sensor as seen in Fig 1-1f. This sensor measures just two or three wavelengths to determine the oxygenation of blood. Similarly, healthy and stressed plants can be discriminated by examining a few narrow spectral bands using specialized viewers called "Nastek glasses" [4].

This work aims to explore and formalize the design of such specialized spectral sensors to encourage their adaptation. In the following chapters, the design of illuminators, sensors, support circuitry and optics for these sensors is discussed. Finally, several such sensors are constructed using these principles to show a proof of their operation.

Chapter 2

System Anatomy

2.1 Introduction

The design of special purpose spectral sensors is focused on its intended use and can vary tremendously from instrument to instrument. That said, many such sensors share somewhat common system topologies and their design is a parameterization of that space.

When presented with a particular spectral use case, it is important to identify the complete parameter space of the design. Obviously the most pertinent parameter is the spectral characteristics of the samples to be discriminated. In order to produce a compelling product, however, many other factors must be considered. These design parameters typically include

- Spectral characteristics of the sample: over which wavelengths can the samples be discriminated?
- Cost: some use cases, e.g., consumer devices, may be highly cost driven, while others, e.g. medical or military applications may be cost tolerant.
- Size: use cases that require a user to transport the device will be inherently more constrained than say a pedestal or kiosk application.
- Power: mobile and autonomous (e.g., robots) devices are sensitive to power consumption, while fixed detectors could be line powered.
- Temperature: large temperature changes can for example, effect the emission and detection wavelengths of solid state detectors. Will the device be fixed in a temperature controlled environment or mobile?

- Vibration: Optical systems, particularly interferometric devices are highly sensitive to positioning errors. Vibrations can reduce the sensitivity of the system or damage it alltogether.
- Detection rate: should the system err on the side of a false positive (e.g., land mine detection) or on the side of a false negative (e.g., mineral prospecting)?

Depending on the particular use case, there may be other confounding design parameters, such as windows or vessels. If the sample is contained in glass, needs to be sensed through a tabletop, or is hot or corrosive, other considerations might be required.

For many of the use cases described in this work, size, cost and power consumption are often minimized. This typically eliminates the suggestion of interferometric and other complicated optical systems. Solid state illumination and sensing devices are the focus of many of the use cases discussed in this work. A thorough discussion of various illumination and detector schemes is included in Chapters 6 and 3. A high level overview of the systems developed in this thesis is discussed in this chapter.

2.2 Passive and Active Illumination

The topology of the spectroscopic systems discussed in this work depends largely upon their particular use case. Many of these cases can be grouped into functional classes. In particular, there are active and passive spectroscopy systems. In an active system, the sample is pumped, or probed, with an illuminator while in passive spectroscopy systems, the sample is illuminated with ambient light. In both systems, the way the sample affects the light is recorded by detectors and used to discover the desired spectroscopic characteristics of the sample.

Passive systems, while intrinsically simpler, lower power, and smaller than active systems, suffer from a principle disadvantage: the ambient illuminator must be accurately characterized to deduce the sample's characteristics. There is a "chicken-and-egg" problem of characterizing spectrum of the illuminance before the spectrum of the sample. There are applications where ambient illumination is necessitated. Take, for example, airborne or space-borne spectroscopy of the Earth's surface. In this case, it is impractical to actively illuminate the sample; ambient illumination by the sun is necessitated. Still, the most striking disadvantage to passive spectroscopy systems, is that they will not operate if there is no ambient illumination.

There are a number of advantages when actively illuminating spectroscopy samples. In particular, by illuminating the sample, influences of background radiation can be mitigated. In addition, if the measurement is heterodyned, the signal-to-noise ratio can be significantly boosted. To do so requires amplitude control of the illumination source. In other schemes, control of the frequency or phase of the illuminator can be used to boost the detection capability of the sensor. Also, if the illumination wavelengths can be controlled, they can be exploited as another dimension when discriminating samples. In many cases, illuminators are far less expensive than detectors, giving economic motivations to multiplex over illumination wavelength rather than detector sensitivity.

2.3 Transmission and Reflection Measurements

This work will consider two broad sample measurement cases. In one case, the sample is illuminated and the spectrum of the reflected light is analyzed to determine the material composition. In the second case, light is shone through the material. The spectrum of the absorbed light is then analyzed to determine material composition.

Generally, transmission measurements are simpler to make than reflection measurements. Transmission measurements are convenient because they afford a fixed and known optical geometry; in most cases, they are resistant to external mitigating factors. Conversely, without a jig or fixture to position the sample, the optical geometry can change from measurement to measurement in the case of reflectance spectroscopy,

In the case of transmission spectroscopy, the material must be transparent. Transmission spectroscopy is typically used for liquid identification and classification. Most solid objects cannot be measured with transmission spectroscopy unless they are dissolved or suspended in a solvent. This tends to limit their applicability to medicine. Nevertheless, there are some locations on the human body which allow transmission spectroscopy, such as the fingertip, earlobe, lip, or web of fingers or toes.

Reflection spectroscopy is typically more convenient then transmission spectroscopy. In reflection spectroscopy, the sample does not need to be transmissive, dissolved or suspended in a liquid, or repackaged. The spectrometer can be brought to the sample instead of the sample being brought to the spectrometer, as is typically done in the case of transmission spectroscopy. Still, reflection spectroscopy is challenging for a number of reasons. In particular, if the surface albedo of the material is altered for inconsequential reasons, the spectroscopic measurement may be adversely affected. For example, on the human body, sweat, makeup, or a suntan would significantly affect the measurement of blood glucose, even though their effect is uncorrelated to the measurement. In addition, most materials are not heterogenous; a sample from one portion of the subject may have a different spectrum than a sample from another portion of the subject. Finally, as previously discussed, without rigid mechanical fixturing, the physical location of the sensor and sample will vary reading to reading, causing inaccuracies in measurements.

2.4 Sensor Topology

In the most generic case, each spectral sensor is composed of either wavelength selective illuminators, detectors or both. The effects of the sample on the light from the illuminator are sensed in the detector. To form a complete system, the light output must be controlled, and the detector amplified and digitized. In addition, some consideration must be made for the mechanical alignment of the illuminator, sample and detector. In the coming chapters, each subsystem is studied then integrated into several specialized spectroscopic sensors.

Chapter 3

Detectors

3.1 Introduction

Light detection by electronic measurement is fundamental in the study of spectroscopy. In a spectroscopic system, detectors are used to compare the transmission or reflection of various wavelengths. Typically the detector is broadband, and the material spectrum is determined by multiplexing the illuminators. In other cases, the illuminator is broadband (e.g. an incandescent light or the sun) whereas the detector is wavelength specific. As explored in this thesis, a combination of narrowband detectors and narrowband sensors are multiplexed to achieve higher material discriminability in some cases. This chapter discusses detector taxonomy and their application to spectroscopic systems.

3.2 Light Dependent Resistors

Light dependent resistors (LDR) or photoresistors, exhibit a resistance which increases with decreasing light intensity. Photoresistors are constructed from bulk semiconductor material. When a photon strikes the photoresistor, electrons can gain enough energy to jump into the conduction band lowering the device resistance.

Despite their low speed response and high noise, photoresistors were traditionally preferred over photodiodes because of their low cost and availability. In recent years, photodiodes have become less expensive, more sensitive, faster, and smaller than photoresistors. Furthermore, hazardous materials regulations have restricted the use of cadmium, an element crucial in LDR construction, further diminishing their use in modern electronics.

3.3 Photomultiplier Tubes

Photomultiplier tubes (PMTs) are often used in ultra-low light applications. A PMT is a vacuum tube which converts incoming photons to electrons. The converted electrons strike a charged electrode inside the tube, releasing even more electrons. Those electrons strike another electrode, multiplying the electrons emitted and the signal. In a PMT, those electrodes are referred to as dynodes.

PMTs are the most sensitive detector, making them ideal for low-light situations. Unfortunately, they are vacuum tubes and, therefore, fragile and subject to mechanical shocks and vibration. They are also large and require high voltages to accelerate electrons to subsequent dynodes. That said, PMTs still find widespread application in scientific instruments such as particle detectors and accelerators.

3.4 Photodiodes

In electro-optic systems, the conversion of light to voltage often occurs in a photodiode. A photodiode, much like a conventional diode, consists of a semiconductor material that is doped to form a PN junction. If the diode is illuminated with a sufficiently energetic photon, a charge carrier is ejected along with the corresponding positive hole. In a photodiode, both electrons and holes are charge carriers. If a charge carrier is ejected in the device's depletion region, it moves towards its respective electrode. The electron is drawn towards the cathode, whereas the hole is drawn towards the anode. This movement of charge carriers produces a photocurrent [1, 18, 23]. Generally, photodiodes fall into one of a few categories:

- PN Photodiodes
- PIN Photodiodes
- Schottky and Metal-Semiconductor Metal (MSM) Photodiodes

- Avalanche (APD) Photodiodes

PN junction photodiodes are the simplest type of photodiode. Similar to conventional PN diodes, PN photodiodes are formed by doping a semiconductor substrate (in this case, silicon) with positive and negative ions to form a junction where P-type silicon and N-type silicon are abutted. When photons impinge on the photodiode, electrons and holes are liberated. If they are not recombined, they are drawn towards their respective electrodes, forming a photocurrent. PN junction photodiodes are the second most common photodiode after PIN photodiodes.

PIN photodiodes, the most common type of photodiodes, are very similar to PN photodiodes. Unlike PN photodiodes, a PIN photodiode has a lightly-doped intrinsic I region between the P and N type regions. The intrinsic region increases the breakdown voltage of the diode, allowing a much higher bias voltage. This drastically increases the speed by reducing capacitance but increases the dark, or leakage, current. In typical applications, however, Johnson thermal noise dominates dark current in the system. Generally, the tradeoff of having to increase bias voltage in order to increase speed is deemed necessary, despite the increased thermal noise that results from doing so. PIN photodiodes have an increased depletion region because of the intrinsic region, causing them to have higher quantum efficiencies and higher sensitivities to longer wavelengths.

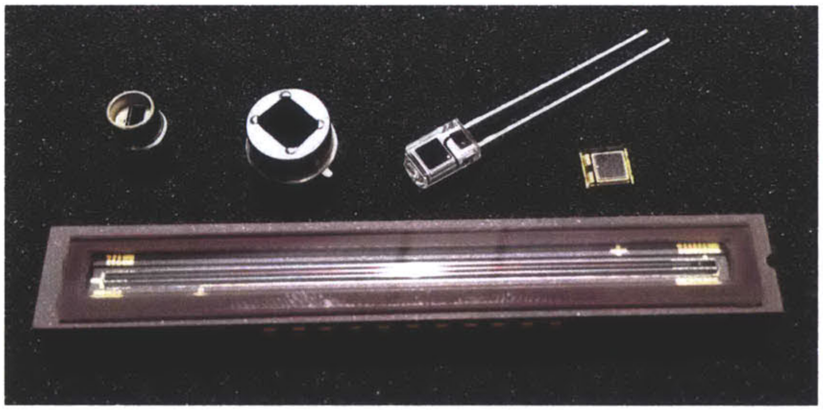


Figure 3-1: Selection of photodetectors. Counterclockwise from the top left, InGaAs photodiode, pyroelectric infrared detector, through hole silicon photodetector, surface mount silicon photodetector, linear CCD array.

Schottky photodiodes are typically formed by sputtering a metal onto an N-type semiconductor. The metal-semiconductor junction forms a Schottkey barrier; the metal becomes an anode and the N-type

Material	Wavelength Sensitivity
Gallium Phosphate	150 nm to 550 nm
Silicon	190 nm to 1100 nm
Germanium	400 nm to 1700 nm
Indium Gallium Arsenide	800 nm to 2600 nm
Lead(II) Sulfide	1 µm to 3.5 µm
Indium Antimonide	1 µm to 5 µm
Indium Arsenide	1 µm to 3.8 µm
Lead(II) Selenide	1.5 µm to 5.2 µm
Mercury Cadmium Telluride	1.5 µm to 12 µm

Table 3.1: Semiconductor materials and their sensitive wavelengths.

semiconductor forms a cathode. The key advantage of Schottky photodiodes is their speed. Schottky diodes have fast recovery times because the metal causes a small charge depletion region at the junction.

Metal-semiconductor metal photodiodes (MSM) are constructed of two Schottky photodiodes, further reducing diode capacitance and increasing speed from rise times of 10 ps to 100 ps. Because photons can interact with metal, Schottkey and MSM photodiodes are sensitive to longer and shorter wavelengths than their PN and PIN counterparts. Schottky and MSM photodiodes are rare, though they do see application in exotic scientific systems.

Avalanche photodiodes (APD) are constructed of a PN junction which triggers an avalanche breakdown in the semiconductor, multiplying the electron hole pairs produced by the impinging photon. These devices, similar to photomultiplier tubes, are used to detect very low light intensities. Unfortunately, because of their avalanche mechanism, they have increased dark current. Similar to Schottky and MTM photodiodes, APDs see little application outside of specialized equipment.

The wavelength sensitivity of a photodiode is determined by the band-gap of the semiconductor it is constructed from. A list of semiconductor materials and their wavelength sensitivity is shown in Table 3.1. Typically, photodiodes are constructed from silicon. Silicon is a convenient material because it is widely understood, commonly employed in the semiconductor industry, and has a band-gap in the visible and near-infrared regime. To construct photodiodes sensitive to other wavelengths, other semiconductor materials are employed [20]. In particular, indium gallium arsenide is often used in near-infrared telecommunications photodiodes and LEDs and mercury cadmium telluride is commonly used in far-infrared, or thermal, imaging systems.

3.5 Photodiode Selection

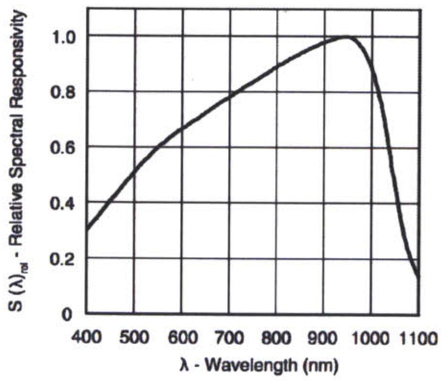
Since most spectroscopic applications have low-photon count, require low-noise precision measurements, and are operated at fairly low speeds, large area photodiodes are desired. Thankfully, selecting photodiodes is a simpler task than selecting LEDs (as seen in Appendix A). With a few simple criteria, the challenge of photodiode selection becomes rather manageable. In this work, the following criteria were used when selecting photodiodes. The selected photodiodes must be:

- Inexpensive: less than \$3.50 each
- Sensitive: an active area between 5 mm^2 to 10 mm^2
- Broadband: responsive to wavelengths from 400 nm to 1100 nm
- Low noise: less than 3 nA dark current
- Available: must be buyable in single unit quantities

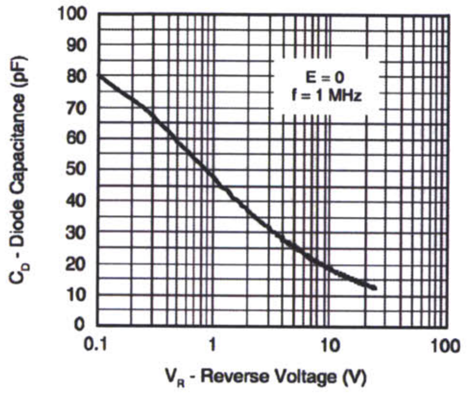
Other features which are desired but not essential include: increased blue sensitivity, convenient mechanical package, and a rise time less than 50 ns.

Unfortunately, simply comparing the manufacturer specifications does not always reveal the true characteristics of the photodiode. In particular, measurement of dark current, rise time, and wavelength sensitivity can vary depending on manufacturer test setup and their interpretation of the results. As is discussed in Chapter 4, the dark current in the device increases and the rise time decreases as the bias voltage is increased. As shown in Fig 3-2b, the capacitance on voltage is plotted for the Vishay TEMD5080, a photodiode used in many experiments. As the reverse voltage is increased, the diode capacitance decreases, increasing its frequency response. The diode breaks down, or starts conducting, when the reverse voltage exceeds 25 V.

Similarly, the wavelength sensitivity of the photodiode is dependent on the manufacturer's interpretation of the responsivity curves. The manufacturer specifies the photodiode minimum and maximum wavelength sensitivity by placing a cutoff at some percentage of the maximum response. These are arbitrary, with most manufacturers specifying a cutoff between 10% to 15%. In order to mitigate differences in data interpretation, careful examination of photodiode data-sheets is required. Fig 3-2a reproduces



(a) Relative spectral sensitivity.



(b) Diode capacitance vs. reverse voltage.

Figure 3-2: TEMD5080 silicon PIN photodiode spectral sensitivity and diode capacitance plots from its data sheet ??.

the spectral response of the Vishay TEMD5080. Since the TEMD5080 is a silicon photodiode, it is sensitive from approximately 350 nm to 1100 nm.

Another factor affecting wavelength responsively is its package material. Some photodiodes, intended for infrared communication and detection, have narrowband or lowpass filters to reduce interference from undesired wavelengths. Other detectors are filtered in such a way to approximate human eye wavelength responsivity. Some photodiodes are said to have increased blue response - often, this is actually a filter which *reduces* red response.

Typically detectors are packaged in transparent epoxy which minimally affects wavelength responsivity. By comparison, some devices are hermetically packaged in metal cans with glass or quartz windows to achieve the ultimate bandwidth. Some experimenters even suggest cutting the can open to expose the bare semiconductor die to further increase frequency response [26].

In table 3.2, a selection of prospective photodiodes are compared and organized by active area. All photodiodes on the table are silicon diodes, and as a result, have a similar wavelength sensitivity of roughly 400 nm to 1100 nm. The green colored cells denote the photodiodes with the most desirable characteristics: a unit cost of less than \$3.50 USD, an active area between 5 mm^2 to 10 mm^2, and a dark current i_{dark} less than 3 nA.

With these considerations, the Vishay TEMD5080X01 was selected for use in the generic spectroscopic

Manufacturer	Part #	Price	Area [mm^2]	λ_{min} - λ_{max} [nm]	t_{rise} [ns]	Max V_{rev} [V]	i_{dark} [nA]	Package
Thorlabs	FDS010	\$42.80	0.82	200 - 1100	1	25	2.5	Can
Advanced Photonix	PDB-C142	\$2.38	4.1	400 - 1100	50	100	5	T1 3/4
Vishay	VBP104S	\$1.01	4.4	400 - 1100	100	60	2	SMT
Vishay	TEMD5020X01	\$1.53	4.4	400 - 1100	100	60	2	SMT
Opto Diode	ODD-5W	\$11.89	5	300 - 1100	10	60	1	Can
Osram	SFH206K	\$1.06	7	400 - 1100	20	32	2	Leaded
Osram	BPW 34 S-Z	\$1.35	7	400 - 1100	20	32	2	SMT
Advanced Photonix	PDB-C160SM	\$4.87	7	450 - 1100	20	32	2	SMT
Osram	BPW34	\$0.87	7.5	430 - 1100	20	60	2	Leaded
Vishay	VBPW34S	\$1.08	7.5	400 - 1100	100	60	2	SMT
Vishay	TEMD5010X01	\$1.61	7.5	400 - 1100	100	60	2	SMT
Osram	BPW46	\$0.95	7.5	400 - 1100	100	60	2	Leaded
Advanced Photonix	PDB-C171SM	\$3.15	7.67	400 - 1100	20	60	4	SMT
Vishay	TEMD5080X01	\$1.82	7.7	350 - 1100	40	25	2	SMT
Advanced Photonix	PDB-C156	\$1.30	8	400 - 1100	15	50	2	Leaded
Advanced Photonix	PDB-C158	\$3.44	9	400 - 1100	50	50	2	Leaded
Opto Diode	ODD-12W	\$15.98	12	400 - 1100	15	60	3	Can
Thorlabs	FDS100	\$13.10	13	350 - 1100	10	25	20	Can
Opto Diode	ODD-15W	\$13.98	15.8	400 - 1100	20	60	4	Can
Advanced Photonix	PDB-V107	\$35.82	17.92	350 - 1100	13	75	0.4	Leaded
Advanced Photonix	PDB-C107	\$21.64	17.92	350 - 1100	13	100	150	Leaded
Opto Diode	ODD-42WB	\$16.99	42	400 - 1100	30	60	11	Can

Table 3.2: Sampling of photodiodes considered sorted according to increasing active area. Cells are shaded green if they meet the desired specifications; if their unit cost is less than \$3.50 USD, have an active area between 5 mm^2 to 10 mm^2, and exhibit dark current i_{dark} less than 3 nA. Generally all photodiodes meet or exceed wavelength sensitivity requirements. For most experiments, the Vishay TEMD5080X01 was selected.

sensor. The TEMD5080 meets or exceeds all desired criteria. It has a wavelength sensitivity from 350 nm to 1100 nm, a large active area of 7.7 mm^2, a fairly quick 40 ns rise time, a dark current of only 2 nA, and the ability to be biased to 25 V. Additionally, the TEMD photodiode is available in a convenient surface mount package and, at the time of this writing, costs only \$1.31 in single unit quantities.

3.6 Light Emitting Diodes as Detectors

Light emitting diodes are formed of PN junctions tuned to a particular emission wavelength. As such, they can be pressed into service as photodetectors. There are a few features which set LEDs apart from conventional PN photodiodes. Since LEDs are tuned to emit at a particular wavelength, they form wavelength specific detectors. LEDs, when used as a photodiode, are sensitive to wavelengths shorter than, or more energetic than, their emission wavelength. LEDs are sometimes encased in colored epoxy to filter their sensitive wavelength further. In addition, multicolored LEDs can be used to discriminate several wavelengths in a single package. Since LEDs are designed for emission rather than detection, the semiconductor die is typically very small and their output optics (e.g., molded lens, light pipe) are not suitable to sensing. Electrically, LEDs are similar to photodiodes, though due to their small die, have much lower capacitance.

Laser diodes can also be useful as photodetectors but have a higher wavelength selectivity than LEDs. They gain this selectivity from tuned anti-reflection coatings on the exit aperture, the band-gap of the laser diode, and the etalon formed by the laser cavity. Furthermore, many laser diodes have an integral photodiode to monitor laser output which can be used as an additional sensor.

3.7 Other Detectors

In recent years, electro-optical conversion has become prominent in a variety of fields and industries. As such, a range of photosensors have been developed.

PN junctions are photosensitive but appear as very high impedance sources of photocurrent. Because of this, they are sometimes combined with other junctions on the same die to increase the output impedance. In a phototransistor, light impinges on the base-collector junction, causing current to flow from the emitter to the collector. The photocurrent on the base-collector junction is then amplified by the transistors gain β. Similarly, in a photo-MOSFET, the light causes the gate to accumulate charge allowing current to flow from the source to the drain, like in the case of an N-fet. Photo-MOSFETs are often employed to switch loads as solid-state relays (SSR) **??**.

Imaging arrays have flourished in light of recent developments in mobile computing and display. Generally, there are two broad classes of focal-plane arrays: CMOS and CCD detectors. A CMOS sensor is an array of photodiodes and amplifiers read out as a matrix similar to memory. A CCD sensor captures buckets of charge in a capacitor array which is shifted out and digitized to form an image. Since all semiconductor junctions are light-sensitive, they are encased in opaque epoxy or ceramic. Some experimenters have exploited this property and used other semiconductors and integrated circuits as light detectors. In particular, RAM has been used successfully as an imaging array [49].

In some applications, thermal measurements are used to convert light to electricity. In such instruments, an element of known thermal coefficient and mass is illuminated. The rate of change of temperature in the element is used to compute the optical power of the light falling on the detector. These devices are called bolometers. This principle can be used to construct micro-bolometers sensitive to mid and far-infrared radiation. When formed into an array, micro-bolometers are used in thermal imaging applications.

3.8 Conclusions

During the course of this work, photodiodes or LEDs were selected as detector elements. Photodiodes and LEDs are rather inexpensive, easy to interface with, and readily available. Most applications considered in this thesis were not photon-limited, obviating the need for a PMT or APD. In practice, with a few exceptions, the Vishay TEMD5080X01 was a suitable detector.

Chapter 4

Photodiode Amplifiers

4.1 Photodiode Basics

As discussed in Chapter 3, in spectroscopic instruments the conversion of light to voltage often occurs in a photodiode. In a photodiode, similar to a conventional diode, a semiconductor material is doped to form a PN junction. If the diode is illuminated with a sufficiently energetic photon, an electron and its corresponding positive hole are ejected. The electron and hole are referred to as charge carriers. If the charge carrier is ejected in the device's depletion region, the carriers move towards their respective electrodes. The electron is drawn towards the photo-cathode while the hole is drawn towards the photo-anode. This movement of charge carriers produces a photocurrent [18,22].

Even if the photon has enough energy, it might not eject a charge carrier into the conduction band. The quantum efficiency QE, or incident photon to converted electron ratio $IPCE$, describes the incidence of photons in the device that produce charge carriers. Since quantum efficiency is a percentage, it does not have a unit. The quantum efficiency for most materials is wavelength dependent QE_λ, and in a photodiode, nonlinear. If the energy of the incident photon cannot excite a charge carrier above the device's band gap, no photocurrent is produced. As previously discussed, this property can be exploited to produce wavelength selective photodetectors using LEDs or laser diodes.

Wavelength dependent quantum efficiency QE_λ is similar to spectral responsively R_λ. This is typically written in the units of A/W, or photocurrent per the energy of incoming photons. To convert from wavelength dependent quantum efficiency to spectral responsively the following equation is employed: wavelength λ is in nm, h is the Planck constant, c is the speed of light, and e is the elemental charge [18,22].

$$QE_\lambda = \frac{R_\lambda}{\lambda} \times \frac{hc}{e} \tag{4.1}$$

In a typical photodiode, such as the Vishay TEMD5080, the quantum efficiency is not given directly. From its datasheet, the device produces 18 µA when illuminated with 1 mW/cm^2 at 400 nm [39]. Since the device has an active area of 7.7 mm, the spectral responsivity $R_{\lambda=400\,\text{nm}}$ is given by

$$R_{\lambda=400\,\text{nm}} = \frac{18\,\mu\text{A}}{7.7\,\text{mm}^2} \times \frac{\text{cm}^2}{1\,\text{mW}} = 0.234\,\text{A/W} \tag{4.2}$$

In addition, the datasheet specifies that when the detector is illuminated with 1 mW/cm^2 at 900 nm light, the device produces 60 µA. Therefore, the spectral responsivity $R_{\lambda=900\,\text{nm}}$ is given by

$$R_{\lambda=900\,\text{nm}} = \frac{60\,\mu\text{A}}{7.7\,\text{mm}^2} \times \frac{\text{cm}^2}{1\,\text{mW}} = 0.779\,\text{A/W} \tag{4.3}$$

An Everlight IR26-21C infrared LED is shone at the TEMD5080 photodetector at its rated maximum power output of 130 mW. If all of the produced photons are coupled into the aforementioned photodiode, perhaps using a lens, reflector, or light guide, the photocurrent i_d produced by the Vishay TEMD5080 is

$$i_d = 0.779\,\text{A/W} \times 130\,\text{mW} = 101\,\text{mA} \tag{4.4}$$

If no collimating optics are used, much of the output of the LED is not captured by the photodiode. If, for example, the photodiode is 1.5 cm away from the LED, the photodiode, with an active area of 7.7 mm, subtends a solid angle of 0.1 sr. Since the light output of the LED is 3.5 mW/sr, the flux incident on the LED is 0.35 mW. In a transmission spectroscopy application, the sample may attenuate 20 dB, reducing the flux on the sensor by another power of ten and leaving just 35 µW incident on the detector. This would produce a theoretical photocurrent i_d of just

$$i_d = 0.779\,\mathrm{A/W} \times 35\,\mu\mathrm{W} = 27.3\,\mu\mathrm{A} \tag{4.5}$$

With more opaque samples, it is possible to see attenuation of 40 dB to 60 dB, decreasing the photocurrent to just 2.73 µA to 0.273 µA. Also, silicon photodiodes are most efficient in the near infrared regime - in the green and further into the blue, the quantum efficiency decreases further (see 4.2). As such, these photocurrents necessitate low impedance amplification for use in spectroscopic systems.

4.1.1 Current to Voltage Conversion

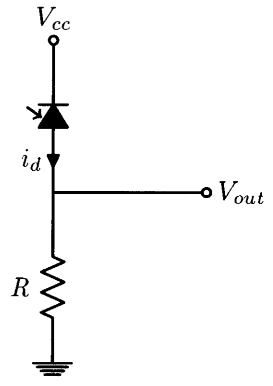


Figure 4-1: A resistor converts current to voltage by Ohm's Law.

The simplest way to convert this current to a voltage is to pass it through a resistor, as seen in Fig 4-1. By Ohm's Law, taking the photodiode photocurrent to be 0.1 µA, the resistor R would have to be quite large.

$$R = \frac{1\,\mathrm{V}}{0.1\,\mu\mathrm{A}} = 10\,\mathrm{M}\Omega \tag{4.6}$$

There are several drawbacks with the approach pictured in Fig 4-1. For one, the 10 MΩ resistor R produces quite a bit of noise. This noise is proportional the resistance R, Boltzman's constant K_B, and the temperature T. The equation [23] for Johnson thermal noise E in resistors per $\sqrt{\text{Hz}}$ is

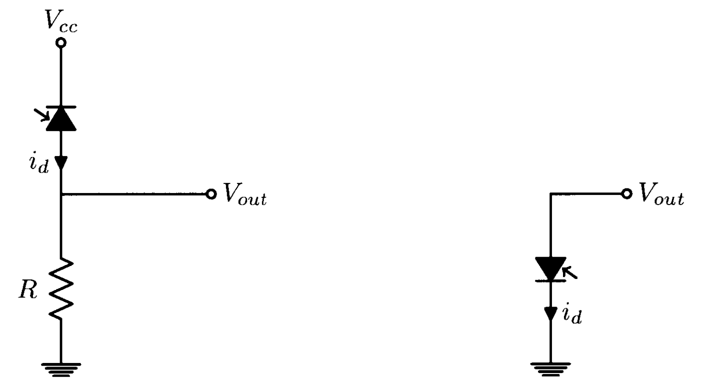
$$E = \sqrt{4K_BTR} \tag{4.7}$$

For a 1 MΩ resistor at room temperature, the resistor contributes around 411 nV/$\sqrt{\text{Hz}}$. This results in a lose-lose situation; as the signal goes down, the thermal noise contributed by the resistor R increases.

Another drawback to this approach is the impedance at the V_{out} node is very high. In Fig 4-1, if $R = 10\,\text{M}\Omega$, the impedance is roughly 10 MΩ, assuming the photodiode dark current gives a much larger apparent resistance. The diode capacitance also significantly contributes to the output impedance at higher frequencies.

In a typical analog-to-digital converter, the input bias current is from 1 µA to 10 µA. The impedance at V_{out} in the circuit pictured in Fig 4-1 is typically too large to interface directly to most analog-to-digital converters, necessitating an amplifier.

4.1.2 Photodiode Operating Modes



(a) Photoconductive mode. (b) Photovoltaic mode.

Figure 4-2: Photodiode operating modes, photoconductive and photovoltaic.

As discussed, there are a number of drawbacks with the current-to-voltage converter illustrated in Fig 4-2a. To increase speed, V_{cc} can be increased to reduce diode capacitance. However, the dark-current in the photodiode will also increased. Also, by Equation 4.7, higher speeds intrinsically increase the thermal noise in the resistor.

The scheme pictured in Fig 4-2a has two sources of noise: the resistor R, discussed in equation 4.7, and the dark current contributed by the photodiode. Dark noise can be modeled as a very large resistor in series with the photodiode. As the bias voltage V_{cc} is increased, the dark current increases.

As pictured in Fig 4-2b, if the bias voltage is eliminated, the intrinsic resistance, or dark current, is proportional only to the photocurrent generated in the device, and, therefore, very small. When the photodiode is not biased, it is said to be operating in "photovoltaic mode". The schematic of a photovoltaic photodiode is shown in Fig 4-2b. Conversely, if the photodiode is biased, as in Fig 4-2a, the photodiode is operating in "photoconductive mode".

In photovoltaic mode (Fig 4-2b), if the voltage at V_{out} is measured using a voltmeter with very high input impedance, the photodiode open-circuit voltage is derived from the device resistance. Unfortunately, the output voltage from this circuit increases nonlinearly with increasing incident illumination.

In photodiode designs, for high-speed, high photon-count applications, photodiodes are used in photoconductive mode. In low-light, high precision applications, photodiodes are connected in photovoltaic mode.

4.2 Photodiode Amplifiers

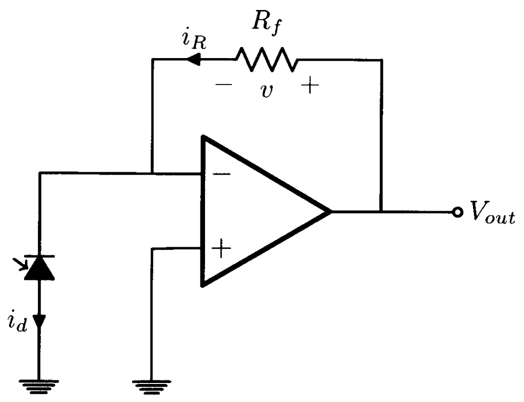


Figure 4-3: Simple transconductance photodiode amplifier.

To completely eliminate the dark current in the device, both terminals of the device are held at the same potential. A simple way to do this is with an operational amplifier. In Fig 4-3, the non-inverting terminal of the op-amp is connected to ground. The inverting terminal of the op-amp consequently forms a "virtual ground," connecting both terminals of the photodiode to the same potential.

Assuming that the inverting and non-inverting inputs of the operational amplifier have infinitely large input impedance, no current flows into those terminals, as illustrated in Fig 4-3. Therefore, the photocurrent generated in the photodiode i_d is equivalent to the current i_R in the feedback resistor R_f.

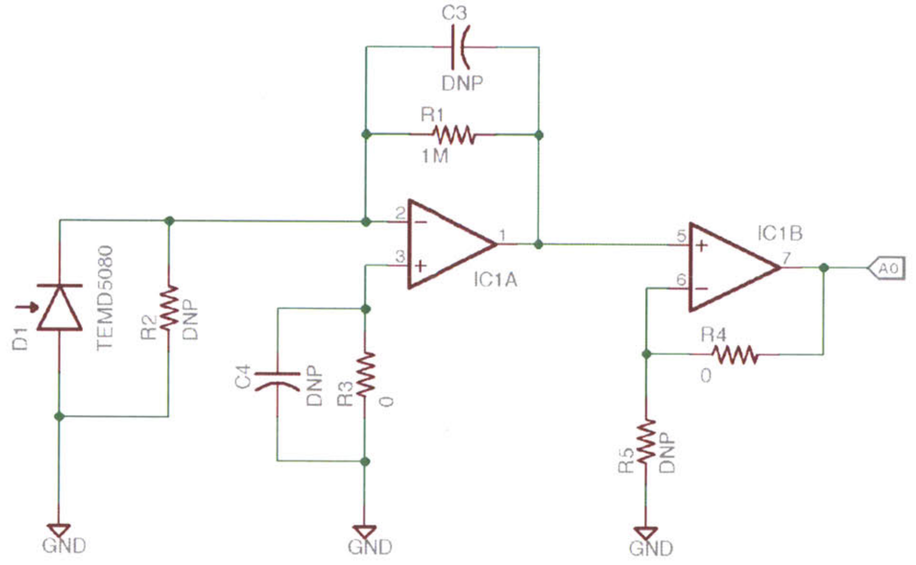


Figure 4-4: Improved and compensated photodiode amplifier.

The operational amplifier servos the voltage on its output to force the voltage applied to its different inputs to be as close to equal to possible. As the current in the photodiode i_d increases, the current through the feedback resistor i_R also increases. This induces a voltage drop v_R across the resistor R_f. Since the resistor is connected to a virtual ground, as the photocurrent increases, the output voltage also increases. This forms an op-amp transimpedance amplifier.

In a transimpedance amplifier, the output voltage V_{out} is simply related to the input current i_d by the following equation

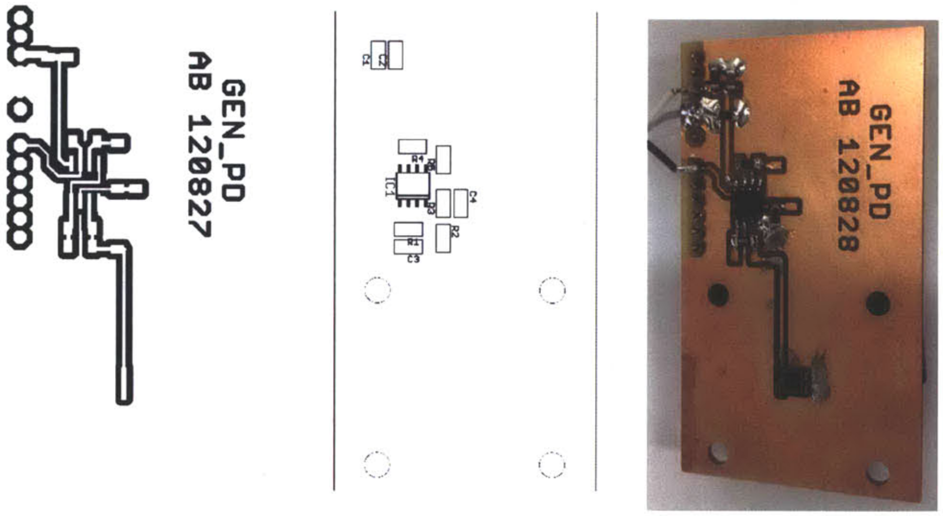
$$V_{out} = i_d \times R_f \tag{4.8}$$

From the previous calculations, the expected photocurrent is on the order of 10 μA. From equation 4.8,

R_f is chosen to be

$$R_f = \frac{5\,\mathrm{V}}{10\,\mu\mathrm{A}} = 500\,\mathrm{k\Omega} \tag{4.9}$$

Since the photodiode has such a large die capacitance, the amplifier feedback lags output. This can lead to clipping and instability. To compensate for the die capacitance, a capacitor is sometimes added in parallel with the resistor R_f. In addition, the input semiconductor devices of the op-amp can exhibit drift and cause a small mismatch in input bias currents. This manifests itself as a DC bias in the output signal. It can be easily corrected with a large 1 MΩ to 10 MΩ resistor in parallel with the photodiode. In addition, some stability problems are compensated with passive components between the non-inverting op-amp input and ground. These and other modifications to the base schematic in Fig 4-3 are shown in Fig 4-4.



(a) Top layer traces and pads.

(b) Placement drawing.

(c) Fabricated and populated board in its base configuration.

Figure 4-5: Diagram of traces and cutout layer, fabricated and populated circuit board.

The improved and compensated photodiode amplifier shown in Fig 4-3 was fabricated in house similarly to other circuits discussed throughout this work. In particular, the schematic was captured and circuit board artwork was designed in EagleCAD. The printed circuit board was designed to have a minimum of 10 mil spacing and 10 mil traces. The top layer traces and pads are illustrated in Fig 4-5a. The placement diagram to guide parts placement on the test circuit, is shown in Fig 4-5b. A photo of the completed test fixture is shown in Fig 4-5c. Due to the high impedance and low current signals present in many parts of the board it is important to carefully wash the circuit to remove contaminants and fluxes. Residue flux left on the circuit can present as small as a $2\,\mathrm{M\Omega/mm^2}$ resistance [37], significantly affecting feedback resistances. The boards were washed in 93% isopropyl alcohol and mechanically agitated to remove contaminants.

4.3 Photodiode Amplifier Performance Evaluation

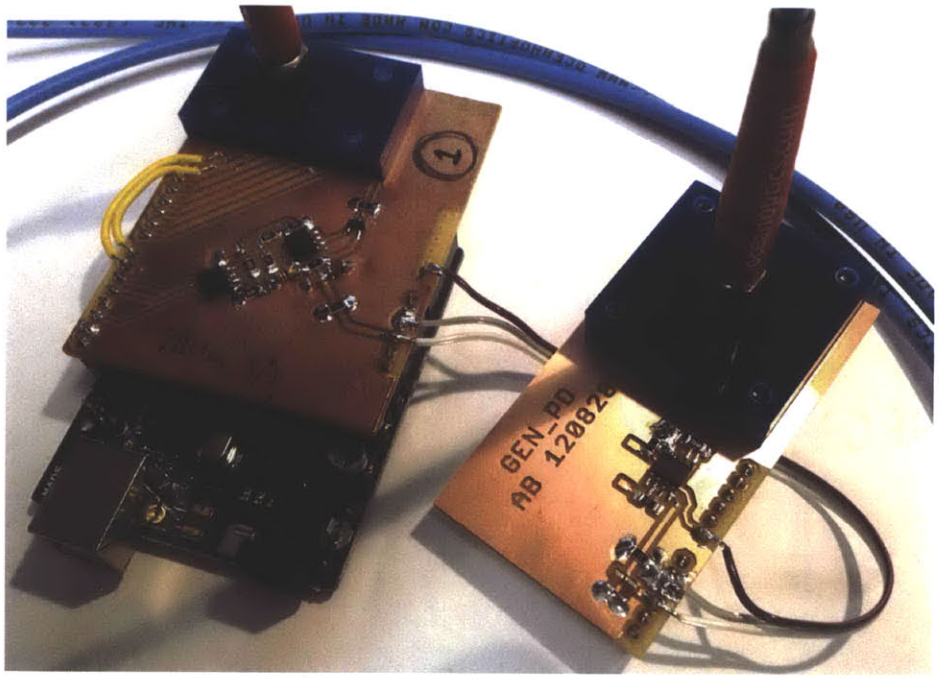


Figure 4-6: General purpose photodiode and amplifier connected to a programmable illuminator for characterization and testing.

To evaluate the design pictured in Fig 4-5, the fabricated circuit was connected both electrically and optically to a programmable illuminator. The programmable illuminator, discussed in Chapter 5, was modified to digitize analog measurements when directed via its serial port. A laser cut acrylic fixture was constructed to firmly couple a SMA connected 600 μm fiber to both the photodiode and LED. Both the general purpose photodiode and illuminator fixture were electrically modified to allow one Arduino to communicate with both the illuminator shield and the general purpose photodiode. Full schematics and firmware are reproduced in Appendix D.

Chapter 5

Programmable Current Sources

5.1 Introduction

Many of the illumination sources used throughout the course of this thesis are semiconductor devices (e.g., LEDs, laser diodes). When the device is forward biased, electrons recombine with holes in the device releasing photons. The device is forward biased when its anode (P-type doped semiconductor) is biased higher than its cathode (N-type doped semiconductor). This recombination, where electrons drop into holes at a lower energy level, occurs with a constant voltage drop. Thus, by conservation of momentum the emitted photon is fairly monochromatic. They are then, by definition, current controlled devices. As the number of electrons flowing through the device increases, the number of photons increases. As the current is varied through the device, the voltage drop remains constant, and the brightness either increases or decreases.

It is desirable to control the current flowing through the device to vary the output brightness. Varying the amplitude can be useful to compensate for temperature or process variations when heterodyning the signal, such as for use with a lock-in amplifier. This is also useful to increase the dynamic range of the sensor, such as in the event the maximum illumination brightness overwhelms the sensor. This chapter describes the design of several current source topologies, their tradeoffs, performance and practical application.

5.2 Simple LED Current Control

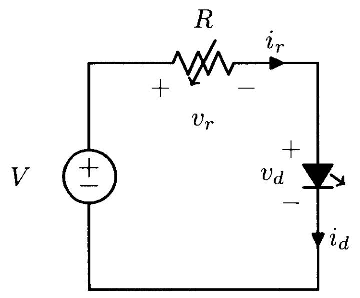


Figure 5-1: Simple LED current control with variable resistance.

The simplest way to control the current flowing through a LED is to use a series resistor, as shown in Fig 5-1. In such a configuration, the voltage in the loop, by the Kirchhoff Voltage Law, is

$$V = v_r + v_d \tag{5.1}$$

Since LEDs are constant voltage devices, the voltage drop v_r is also constant. However, the current through the resistor i_r is proportional to its resistance by Ohm's Law. Substituting this into the previous equation gives

$$V = i_r * R + v_d \tag{5.2}$$

Since this is a closed system, Kirchhoff's Current Law states the current through the resistor i_r equals the current through the diode i_d. Thus, to increase the current i_d in the diode, the resistance R should be decreased in this circuit. Simplifying the current in the diode i_d is

$$i_d = \frac{V - v_d}{R} \tag{5.3}$$

Another approach, to be explored in the next section, is to vary the voltage V while keeping the resistance R constant as illustrated in Fig 5-2.

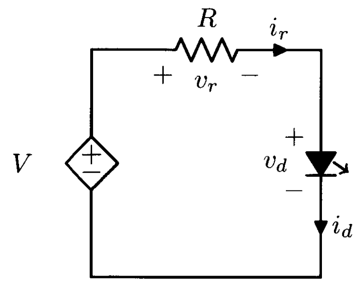


Figure 5-2: LED current control with variable voltage.

The current through the LED in the circuit in Fig 5-2 is governed by the same equation as the circuit in Fig 5-1, except the voltage source V, rather than the resistance R, is varied.

$$i_d = \frac{V - v_d}{R} \tag{5.4}$$

5.3 Programmable Linear Current Supplies

In order to have programmable control of the current flowing through the device, the previous circuit would need to have a programmable voltage source. This controllable voltage source would then use a resistor to convert its voltage to a controllable resistance (as in Fig 5-2).

As illustrated in Fig 5-3), an example of this is a micro-controller which communicates with a Digital to Analog Converter (DAC) and is buffered by an operational amplifier (op-amp) configured as a voltage follower. A resistor and LED are in series with the output of the voltage follower. The resistor converts the variable voltage to a variable current as in Fig 5-2.

Many of the circuits described in this chapter use a Digital to Analog Converter (DAC). A typical choice for such a DAC is the Texas Instruments (TI) TLV5638. The TLV5638 is a dual, 12-bit SPI digital to analog converter [46]. A simplified schematic of one converter is illustrated in Fig 5-3. The converter is modeled as a variable resistor configured as a voltage divider which can be adjusted in 4096 (2^12) steps. The high side of the voltage divider is connected to either an internal (Zener) voltage reference or an external reference V_{ref}. In this particular DAC, the output is multiplied by two.

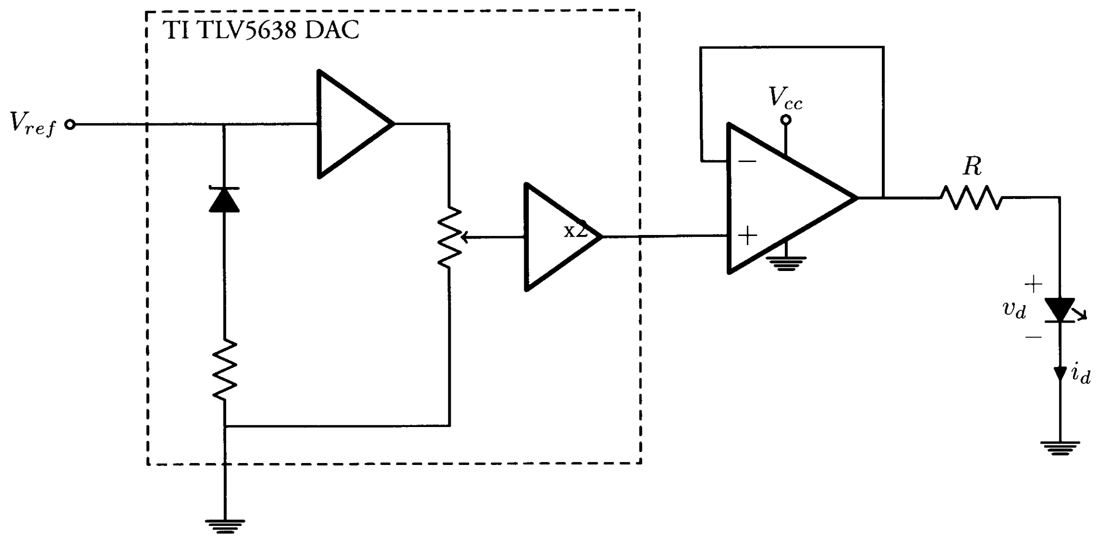


Figure 5-3: Voltage controlled programmable current source.

There are a number of complications with this approach. For one, the operational amplifiers typically cannot source or sink more than 10 mA. Most LEDs used during the course of this thesis have peak outputs when operated between 10 mA and 50 mA. There are some op-amps which are specially designed to drive low-impedance loads and can source or sink in excess of 75 mA, such as line drivers or audio amplifiers. These amplifiers are uncommon, typically more expensive, and often make concessions to achieve such high output currents.

Another complication present in both the DAC and the op-amp are non-ideal near rail operation of amplifiers. In particular, amplifiers become nonlinear near their maximum and minimum outputs, or rails. Some amplifiers, cannot reach their supply rails, while others can. This comes with certain concessions, such as a loss of linearity and speed. The circuit illustrated in Fig. 5-3 shows that if the op-amp does not have rail-to-rail operation, the full current range cannot be achieved. This because the voltage cannot be varied over the full supply range. In the DAC, errors near rail voltages are described as zero-scale error E_{zs}. For the TLV5683, E_{zs} is specified as ± 24 mV. This error, E_{zs}, implies the circuit cannot be off. That is, i_d can never be zero.

The circuit, as illustrated, also does not make full use of the resolution provided by the converter. As drawn, the circuit can provide

$$i_d = \frac{V - v_d}{R} \tag{5.5}$$

Depending on the selection of the resistor value R, the full resolution of the converter will not be used. If the circuit is designed to supply between 0 and 50 mA and we take v_{cc} as 5 V, v_d as 3 V, and R as 100 Ω

$$i_d = \frac{5\,\mathrm{V} - 3\,\mathrm{V}}{10\,\Omega} = 200\,\mathrm{mA} \tag{5.6}$$

If the circuit is only operated with the maximum current of 50 mA and the maximum current available is 200 mA, only a quarter of the converter's resolution will be utilized. This is ordinarily not a problem, as an appropriate resistor could simply be selected to use the entire converter resolution. However, if the diode voltage drop v_d is unknown or variable, the resistance R cannot be preselected. In addition, for an arbitrary drop v_d it may be impossible to select some resistance to optimally use the entire resolution.

The greatest drawback to the circuit proposed in Fig 5-3 is that with varying LEDs, there will be varying voltage drops. Since the voltage drops do not remain constant, the current i_d through the diode will have to be calibrated and configured for each LED. This circuit requires that the LED is characterized before use, making it impractical for some applications.

Ideally, a circuit would be constructed such that the current through the device i_d is controlled independent of the device's voltage drop v_d. Particularly, the controller should be designed such that the entire resolution of the converter is used to maximize programmability.

5.3.1 Simple Linear Supplies

All of the designs discussed so far have shared a common theme; voltage is controlled to form a varying current in a resistor. As discussed, there are a number of drawbacks in the preceding designs making them typically unsuitable for use.

Discussion

The following designs mitigate some of the device's drawbacks by sensing voltage drop across a resistor which is proportional to current flow. This voltage directs a simple analog control loop to produce a constant current within the linear regime of the system. Closed loop control is advantageous because the actual current can be controlled through the device.

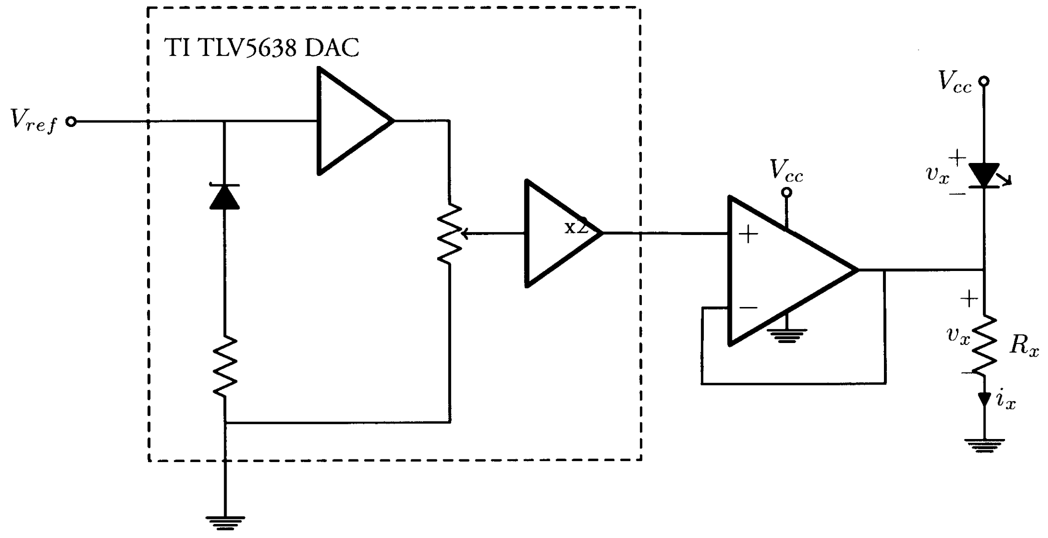


Figure 5-4: Closed loop LED current control.

Fig 5-4 illustrates the simplest example of such a topology. In this design, the current through the resistor R_x is sensed by the inverting input of the op-amp. To set the current through the resistor, we set the voltage on the non-inverting input. The op-amp adjusts its output to bring its two inputs to as close of a potential as possible. By putting a potential on the non-inverting input, the op-amp will drive its output until the two inputs are equal, forcing a voltage drop and, therefore, a current through the sense resistor. The current through the device and resistor i_x is given by this relationship to the voltage applied on the non-inverting input V_+ of the op-amp by the DAC.

$$i_x = \frac{V_+}{R_x} \tag{5.7}$$

The circuit illustrated in Fig 5-4 improves over the circuit drawn in Fig 5-3 because it offers closed, rather than open, loop control over the current through the device. If V_{cc} is greater than the device's forward

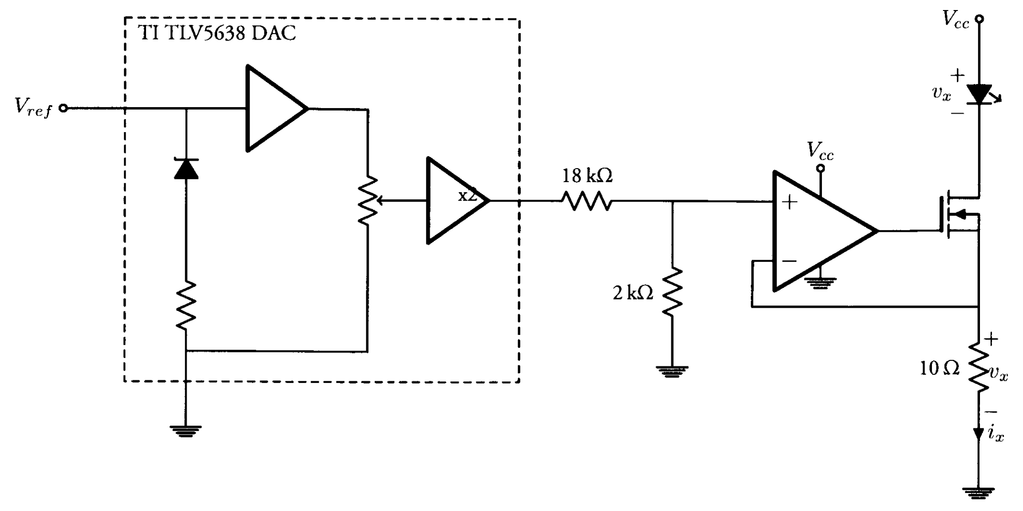


Figure 5-5: Programmable linear constant-current supply with N-channel bypass MOSFET.

voltage drop, the current through the device is independent of the forward voltage drop.

This improved circuit still suffers from many of the drawbacks present in the previous circuit. Since, generally, many of the same components are shared between the two topologies, such as the op-amp and the DAC, limitations associated with those devices are present in this improved circuit as well. Unlike the previous circuit, some of the limitations of this circuit can be mitigated.

The circuit, as drawn in Fig 5-4, can utilize the entire dynamic range of the converter without careful design, like the circuit shown in Fig 5-3. Typically, a programmable current source with a maximum current of 50 mA is desired. Inserting this value into Equation 5.7, taking V_{cc} as 5 V, and solving for the resistor R_x will give the value which optimally uses the available resolution of the converter.

$$\frac{5\,\mathrm{V}}{50\,\mathrm{mA}} = 100\,\Omega \tag{5.8}$$

In this example, the resistor value worked out, but, typically this value is not so convenient. Furthermore, this resistor is a sense resistor which needs to be capable of dissipating - in this case, recalling $P = I^2R$,

$$50\,\mathrm{mA}^2 * 100\,\Omega = 0.25\,\mathrm{W} \tag{5.9}$$

Due to the higher power dissipation requirements, typically R_x is fixed, and the DAC's output is scaled appropriately to use its full resolution. The previously discussed zero scale error E_{zs} can be minimized if the dividing and sense resistors are carefully selected. After some study, it was found that if the voltage scaling is carefully split between the voltage divider and the sense resistor the zero scale error can be minimized.

There is still the lingering issue of current output. As previously discussed, op-amps typically cannot sink or source more than 10 mA, and for the purposes of this study, at least 40 mA are desired. A solution to increase the current output of the circuit in Fig 5-4 is to add a pass transistor. This transistor, in this case, a N-Channel MOSFET, and the scaling resistors are shown in the modified schematic in 5-5. When V_{Ref} is chosen as $V_{cc}/2 = 5\,\text{V}$, this circuit is designed to supply from zero to 50 mA using the full twelve bit resolution of the converter.

Prototype Development

To verify proper operation, this circuit was first prototyped on a solder-less breadboard. In the prototype, as shown in Fig ??, a precision multi-turn potentiometer was substituted for the TI TLV5683 digital-to-analog converter.

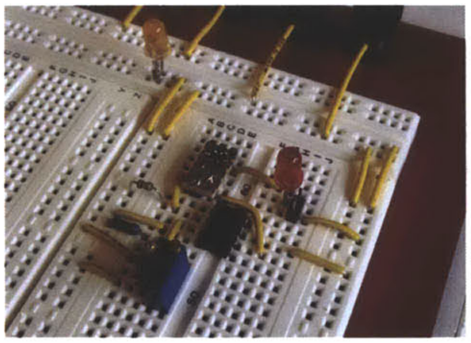
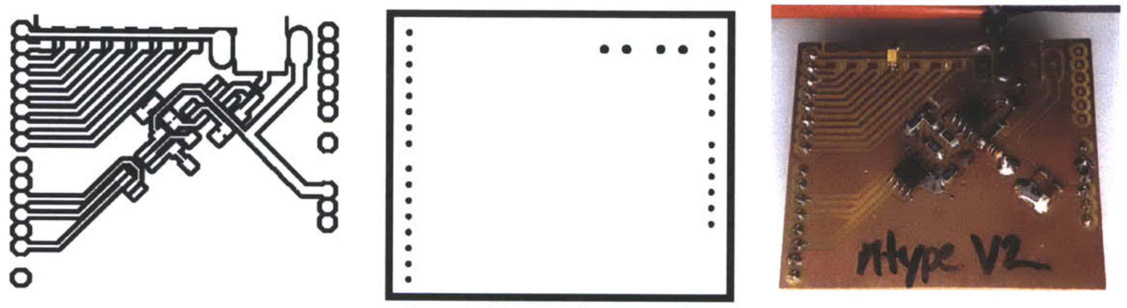


Figure 5-6: Circuit mocked up on a solder-less breadboard.

Once proper operation of the current source was verified, the schematic was captured into EagleCAD and an Arduino compatible shield was fabricated to further characterize the performance of the circuit. The shield was attached to the Anduino, described in Chapter C. The schematic of the prototype is illustrated in Fig 5-8.

This prototype was developed to characterize the true current output of the device and synthesize a design suitable for many sensing applications. There are several features found in this circuit that makes it suitable for prototype development and could function as a proof of concept for driving solid state illumination for spectroscopy. Some features of note are programmable current sinking (via the circuit discussed previously in Fig 5-8), test points to measure current output, and pads for multiple devices.



(a) Top layer traces and pads. (b) Holes and board outline. (c) Fabricated and populated board.

Figure 5-7: Diagram of traces and cutout layer, fabricated and populated circuit board.

A circuit board electrically consistent with the schematic shown in Fig 5-8 was then designed in EagleCAD. The circuit was designed following 10/10 mil rules, for a single sided PCB. All components are either through hole, SOIC, or 1206 passives. Once the board was designed, a prototype circuit board was milled out of copper clad FR-1 in-house on a Roland MODELA MDX-20.

The circuit was milled in two steps. First, the traces and pads (as shown in Fig 5-7a) were milled using a 0.4 mm square-shoulder center-cutting endmill, as shown in Fig 5-7a. Next, the board outline, slots, tabs and drills were cut using a 0.8 mm square-shoulder, center-cutting endmill. A photo of the completed circuit board is shown in Fig 5-7c. In the photo, the circuit board is mounted to an Anduino and connected to an Agilent 34401A to measure its true current output.

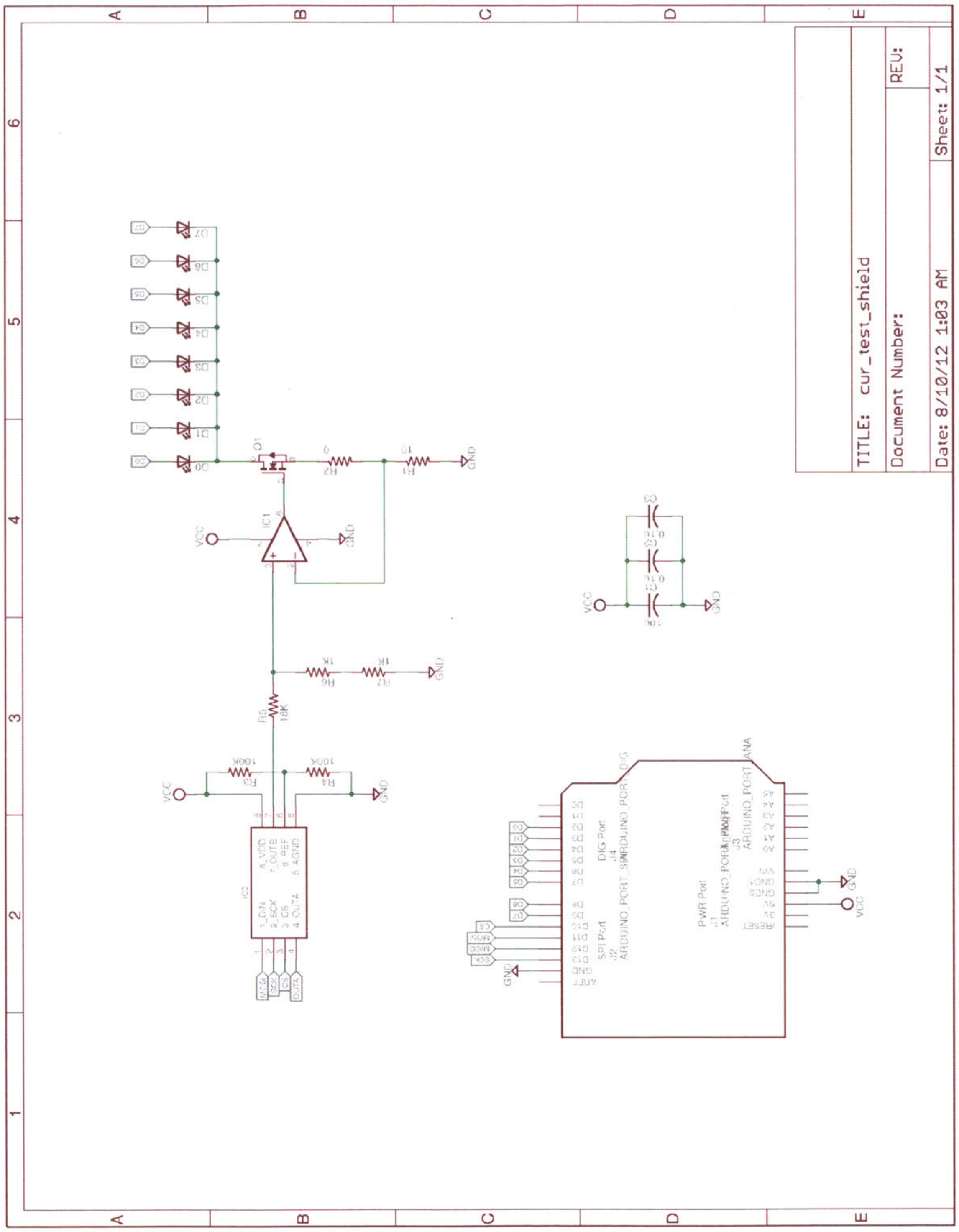


Figure 5-8: Schematic of prototype constant current supply.

Prototype Characterization

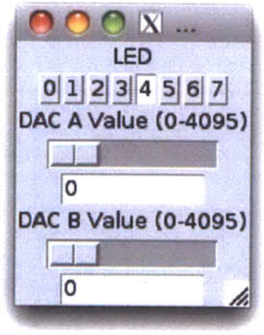


Figure 5-9: Screenshot of the control GUI.

The circuit described in Fig 5-8 and its corresponding layout shown in Fig 5-7a and Fig 5-7c is characterized by affixing the board to an Arduino clone, an Anduino (described in Appendix C).

The following firmware for the Anduino listens on the serial port for a DAC address and value. When it receives a valid packet, it sets the output of the DAC to its commanded voltage. This voltage, in turn, corresponds to a current that is detected as a voltage through the 10 Ω sense resistor ($R1$ in Fig 5-8).

Assume that V_{cc} is exactly 5 V, $R1$ is exactly 10 Ω, V_{ref} is exactly $V_{cc}/2$, and recall that the TI TLV5638 digital-to-analog converter has twelve bits of resolution. For some code C, the current through the device i_x is given by

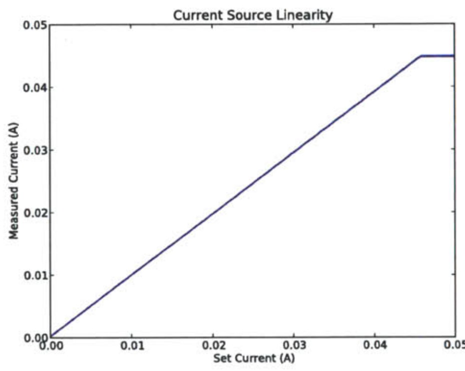
$$5\,\mathrm{V} \times \frac{C}{2^{12}} \times \frac{2\,\mathrm{k}\Omega}{18\,\mathrm{k}\Omega + 2\,\mathrm{k}\Omega} \times \frac{1}{10\,\Omega} = \frac{C}{10 \times 2^{13}} \tag{5.10}$$

The firmware in Appendix E uses the Arduino environment and libraries for serial communication with the computer and SPI for communication with the digital-to-analog converter. The firmware makes no assumptions about analog circuitry connected to the output of the digital-to-analog converters and is only intended to control their respective output voltages.

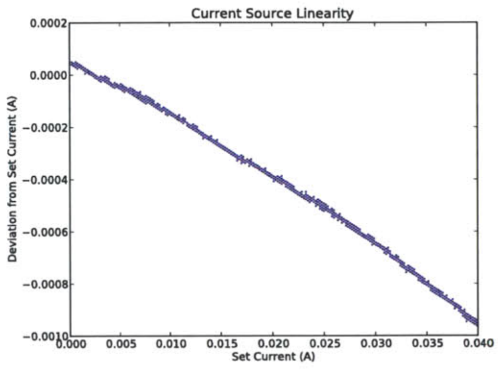
A simple GUI using TkInter and Python was authored to test the output of the digital to analog converter. The GUI (shown in Fig 5-9) has a slider and direct input for each channel of the digital to analog converter. This GUI makes no assumptions about any analog circuitry connected to the output of the digital-to-analog converters and is only intended to control their respective output voltages.

This GUI was used extensively during fine tuning of component parameters and illumination values throughout this thesis. The source for this simple GUI can be found in Appendix E.

Performance Evaluation



(a) Measured current on set current of the circuit shown in Fig 5-8.



(b) Zoomed portion of measured linearity of circuit shown in Fig 5-8, the circuit is very linear.

Figure 5-10: Experimental results of Fig 5-8 linearity.

Similar to the evaluation performed in Appendix B, the current source was characterized to establish its linearity with respect to its set-point. To do this, the current source was programmed to step through its range in 256 steps. Assuming all components are perfectly matched to their value, each step should be 50 mA/256 or nearly 200 μA.

The Python script `lin.py` in Appendix E steps the current source through 256 levels, or every 200 μA from zero to 50 mA fifteen times. The actual current through the device is then measured with an Agilent 34401A and recorded in a CSV file for later processing.

From the recorded currents, the linearity of the code versus the linearity of the current in the device was plotted in Fig 5-10 below using the script below. To understand the code it is worth recalling equation 5.10, which is reproduced below.

$$5\,\mathrm{V} \times \frac{C}{2^{12}} \times \frac{2\,\mathrm{k\Omega}}{18\,\mathrm{k\Omega} + 2\,\mathrm{k\Omega}} \times \frac{1}{10\,\Omega} = \frac{C}{10 \times 2^{13}} \tag{5.11}$$

In this equation, a digital code is converted to some outputted current. In lines 28 and 29, the simplified equation is used to convert the code into an expected current reading.

Two plots, illustrating the linearity of the circuit under real world conditions, are shown below in Figs 5-10a and 5-10b. As previously discussed, these plots were generated using the aforementioned source `lin.py` and `plot.py`. The current source described in Fig 5-8 was used with a Lite-On LTW-150TK white LED.

Upon examination of the plots below, it is apparent the circuit exhibits a high degree of linearity up to around 45 mA. Despite this curious behavior, the circuit is still rather linear within its design specifications as shown in Fig 5-10b. A number of factors may have contributed to this anomaly. The most obvious culprit is the limited amount of current the micro-controller can source per pin. According to the ATMega368 datasheet, each pin can source around 40 mA and the entire device can source 200 mA. It is worth noting that the ATMega368 can sink more current than it can source because N-type devices typically have higher current carrying capacities than their P-type brethren.

There are a number of other sources of error and uncertainty in this test setup. Unfortunately, it is very difficult to directly measure currents. As such, currents are measured indirectly. Techniques for measuring currents include calorimetry, measurement of its intrinsic magnetic field, and measurement of a voltage drop across a known low temperature-coefficient resistor. In the Agilent 34401A, the voltage drop across a known resistance is carefully measured and current is computed. Depending on the measurement range, different current-sense resistors are used. The resistors are carefully selected to have low temperature-coefficients so the heating cause by the current does not affect the resistance, influencing the measured voltage. The simplified measurement schematic is illustrated in Fig 5-11.

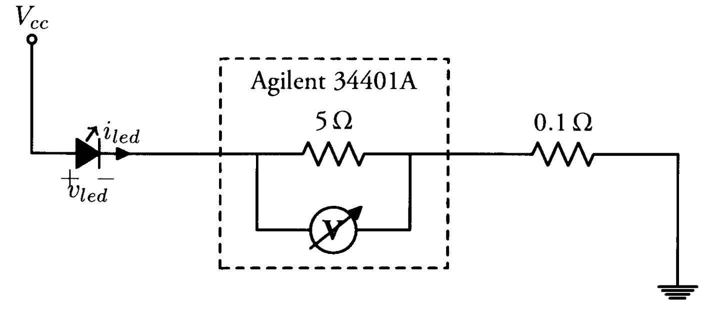


Figure 5-11: Simplified measurement schematic.

According to the Agilent 34401A datasheet [44], in the 100 mA range, the meter presents a 5 Ω load so it can measure the voltage drop across it and compute current consumption. In some situations, this does not present significant errors in the measurement. In others, it can drastically affect measured currents. For example, if 40 mA are dropped across a 5 Ω sense resistor, by Ohm's Law, 200 mV are dropped across the resistor.

According to the Lite-On LTW-150TK datasheet [25], its forward voltage drop v_{led} is 3.8 V. When i_{led} is 40 mA, the voltage drop of the Agilent 34401A is 200 mV. In this case, control circuitry must drive the voltage across the 0.1 Ω sense resistor R_x to 1.2 V. If the minimum R_{DS}, or on resistance, of the MOSFET cannot be made small enough, the circuit may exhibit nonlinearity. That said, in this case, the nonlinearity exhibited most certainly arises from the current carrying capacity of the micro-controller.

Other Topologies

A number of other topologies were explored during the development of this source. Since most micro-controllers can sink more current than they can source, several high-side current source designs were developed.

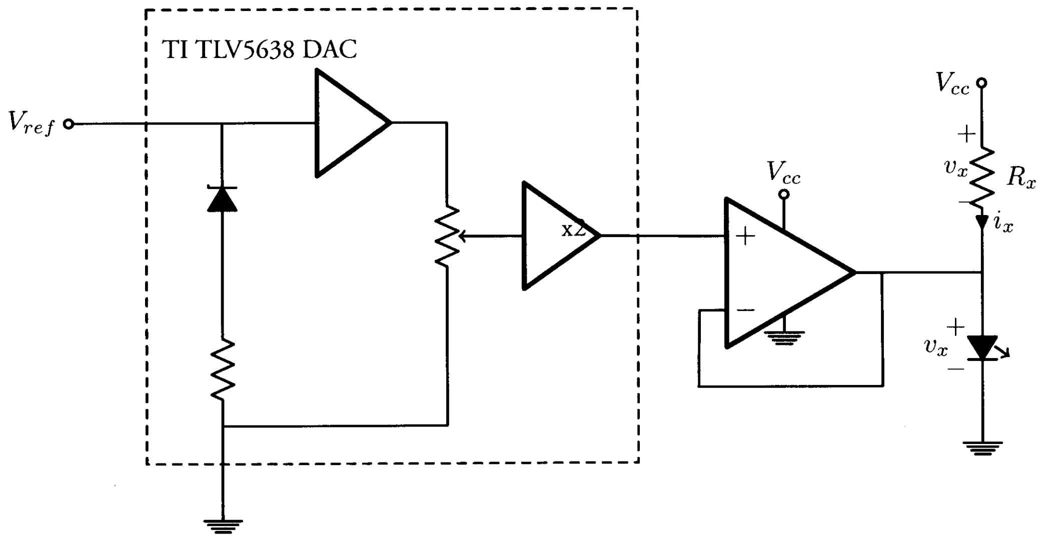


Figure 5-12: High side closed loop LED current control.

An example of high-side close loop current control is an inverted version to the circuit presented in Fig 5-4. In this circuit, shown in Fig 5-12, the sense resistor is on the high side of the LED. To program a

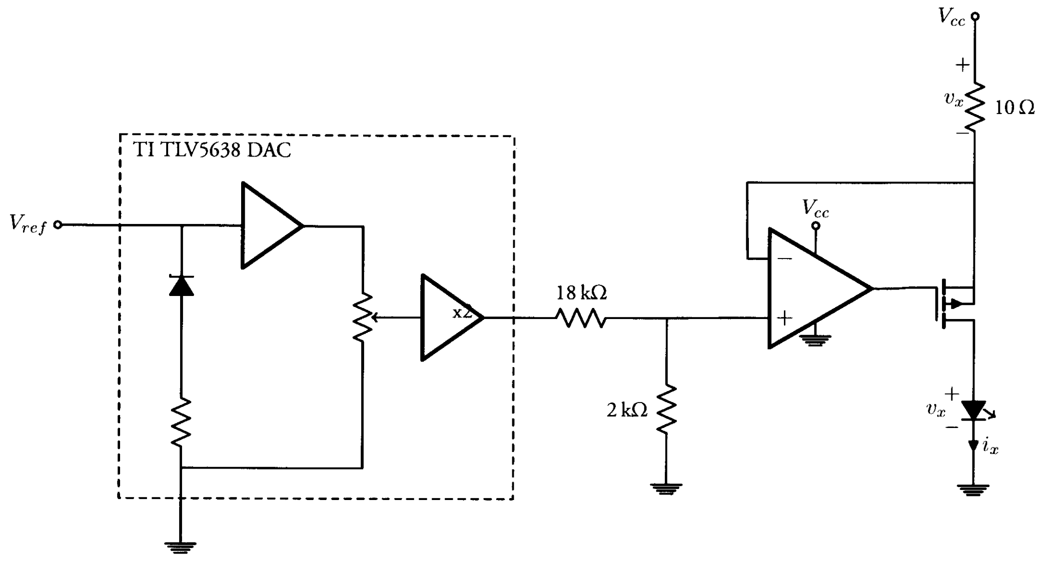


Figure 5-13: High-side close loop current control with pass transistor.

current through the device, a voltage proportionally lower than the supply voltage V_{cc} is applied to the non-inverting input of the op-amp. Since the voltage drop across the sense resistor is proportional to the current through the resistor by Ohm's Law, the current can be programmed by setting the voltage at the op-amp's output node. Unlike the circuit in Fig 5-4, the current through the resistor is inversely proportional to the voltage applied to the non-inverting input and, therefore, present on the output of the op-amp.

With typical op-amps and values of R_x less than $50\,\Omega$, the design current through the device i_x cannot be achieved. Similar to the previous design in Fig 5-5, a transistor is used to bypass the output of the op-amp to increase its current carrying capacity. In this case, because the transistor is on the high side of the load, a P-type, rather than a N-type, MOSFET is employed. This modification is shown in Fig 5-13.

Similar to the previous circuit, an Arduino compatible shield was designed and manufactured to characterize this proposed circuit. The schematic for the high side programmable constant current supply is pictured in Fig 5-14 below. The circuit was routed on a single sided printed circuit board for production in-house using rules similar to prototypes developed elsewhere in this chapter. The circuit has a 10 mil minimum trace width and a 10 mil minimum spacing between traces.

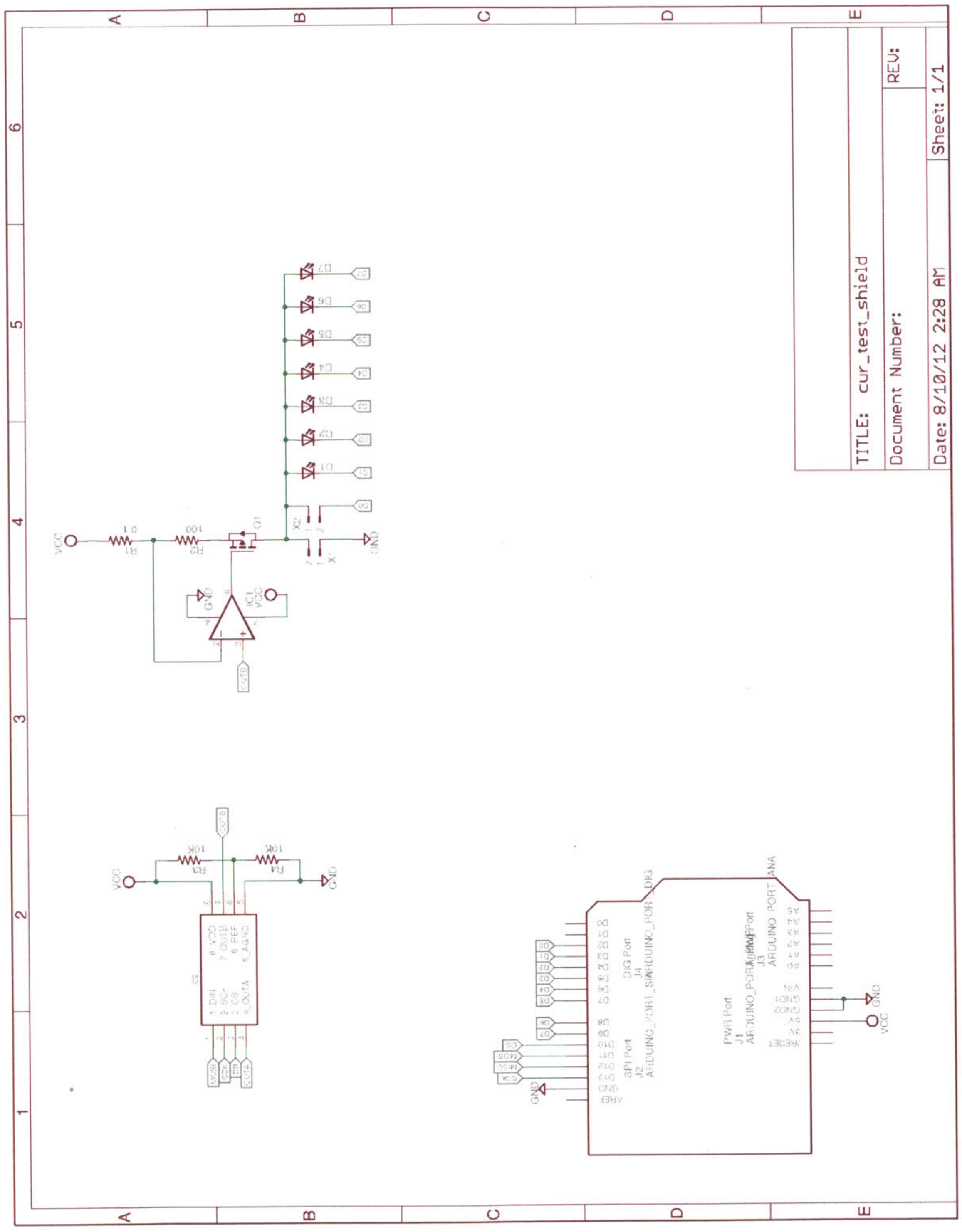
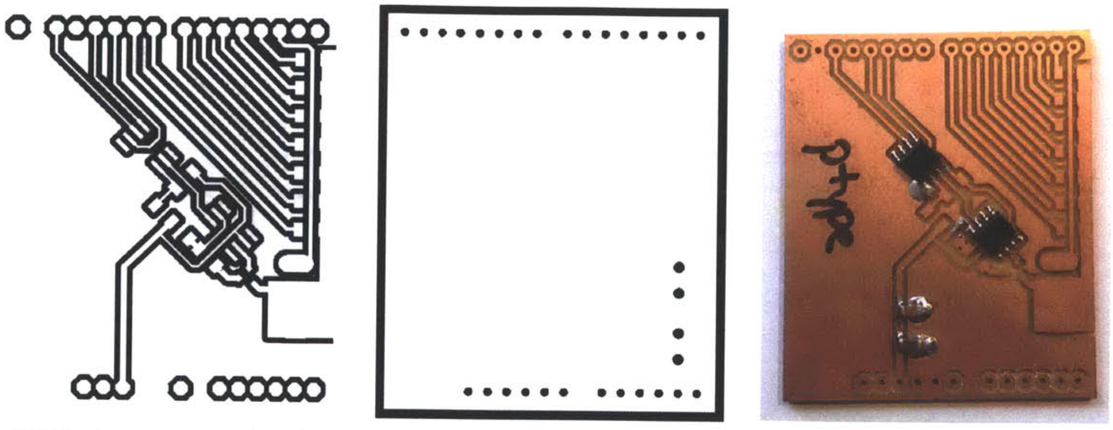


Figure 5-14: High-side programmable current supply.



(a) Top layer traces and pads. (b) Holes and board outline. (c) Fabricated and partially populated constant-current supply.

Figure 5-15: Fabricated and populated high-side linear constant-current supply, top-layer artwork, drills and cutout layers.

The completed single-sided top-layer artwork of the circuit shown in Fig 5-14 is illustrated in Fig 5-15a. The drills and board outline are shown in Fig 5-15b. The fabricated and populated high-side constant-current linear power supply is shown in Fig 5-15c. The high-side supply was evaluated similarly to the low-side supply. Unfortunately, there were some problems with its operation, due to nonlinearities caused by operating both the op-amp and the FET too close to the supply rail. This design was abandoned, as these problems can be resolved using a low-side linear programmable current source, as illustrated in Fig 5-8).

5.3.2 Howland Current Pump

Unlike the supplies discussed elsewhere, the Howland current pump can both sink and source current. This can be useful for characterizing complete device characteristics. For example, in the case of diodes, a Howland current pump can be used to ascertain both the forward and reverse voltage drops. Another application could be the biasing of photodiodes. The Howland current pump can be used to switch the diode's operating mode between forward and reverse bias.

The Howland current pump is interesting both electrically and historically. After being invented at MIT in 1962 by Prof. Bradford Howland [47], the Howland current pump was widely used at both MIT and Philbrick Researches in early analog computers [27].

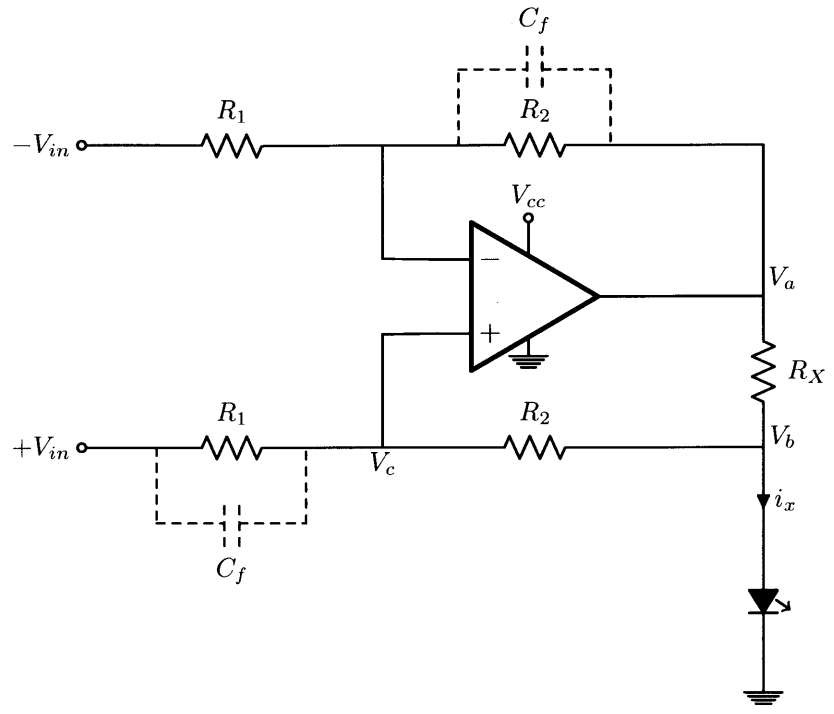


Figure 5-16: Improved Howland current pump.

The current pump, pictured in Fig 5-16, is best understood through inspection. First, the equation for i_x is written with respect to the input voltage $-V_{in}$ taking $+V_{in}$ as ground 0 V.

$$i_x = \frac{V_{in} - V_c}{R_1} = \frac{V_c - V_a}{R_2} \tag{5.12}$$

Next, the voltage V_c is written with respect to V_b.

$$V_c = V_b \frac{R_1}{R_1 + R_2} \tag{5.13}$$

Solving equation 5.12 for V_c and substituting into equation 5.13 gives

$$\frac{R_2 V_{in} + R_1 V_a}{R_1 + R_2} = \frac{V_b R_1}{R_1 + R_2} \rightarrow R_2 V_i n + R_1 V_a = V_b R_1 \tag{5.14}$$

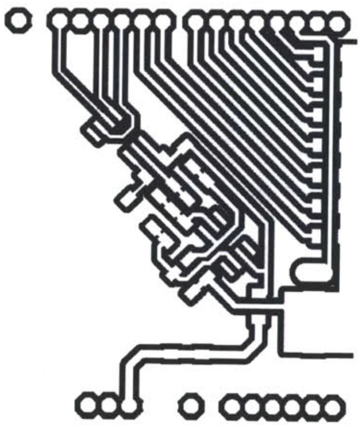
If R_2 is large, it can be assumed that the current flowing through R_x is the same as the current through the load i_x and is given by

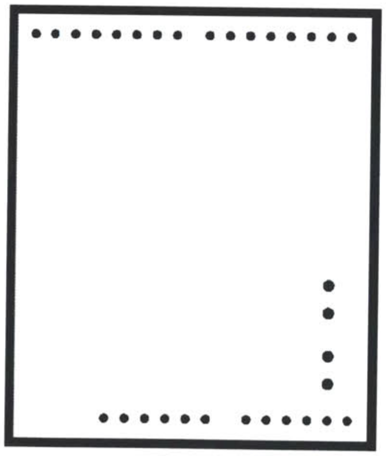
$$i_x = \frac{V_a - V_b}{R_x} \tag{5.15}$$

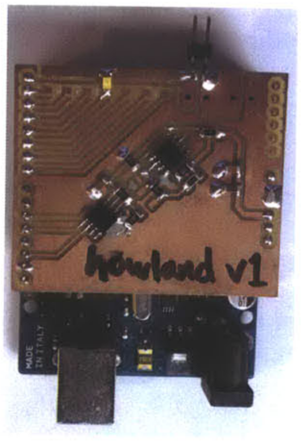
Solving equation 5.14 for $V_a = V_b$ and inserting into the previous equation gives

$$i_x R_x = V_a - V_b = \frac{-R_2}{R_1} V_{in} \rightarrow i_x = \frac{-R_2}{R_1 R_x} V_{in} \tag{5.16}$$

Similar to the other current supply topologies, the Howland current pump was drawn as an Arduino shield to evaluate its performance. The schematic of the evaluation board is show in Fig 5-18. The shield was designed to be manufactured in-house with 10 mil minimum trace width, 10 mil spacing, SOIC, 1206, and SOT-23 parts. The top-layer artwork is shown in Fig 5-17a, the drills and board outline are shown in Fig 5-17b, and the populated and assembled board is shown in Fig 5-17c. In the schematic, drawn in Fig 5-18, capacitors C_f are sometimes required to ensure stability.


(a) Top layer traces and pads.


(b) Holes and board outline.



(c) Fabricated and populated Howland current pump.

Figure 5-17: Fabricated and populated improved Howland current pump, top-layer artwork, drills and cutout layers.

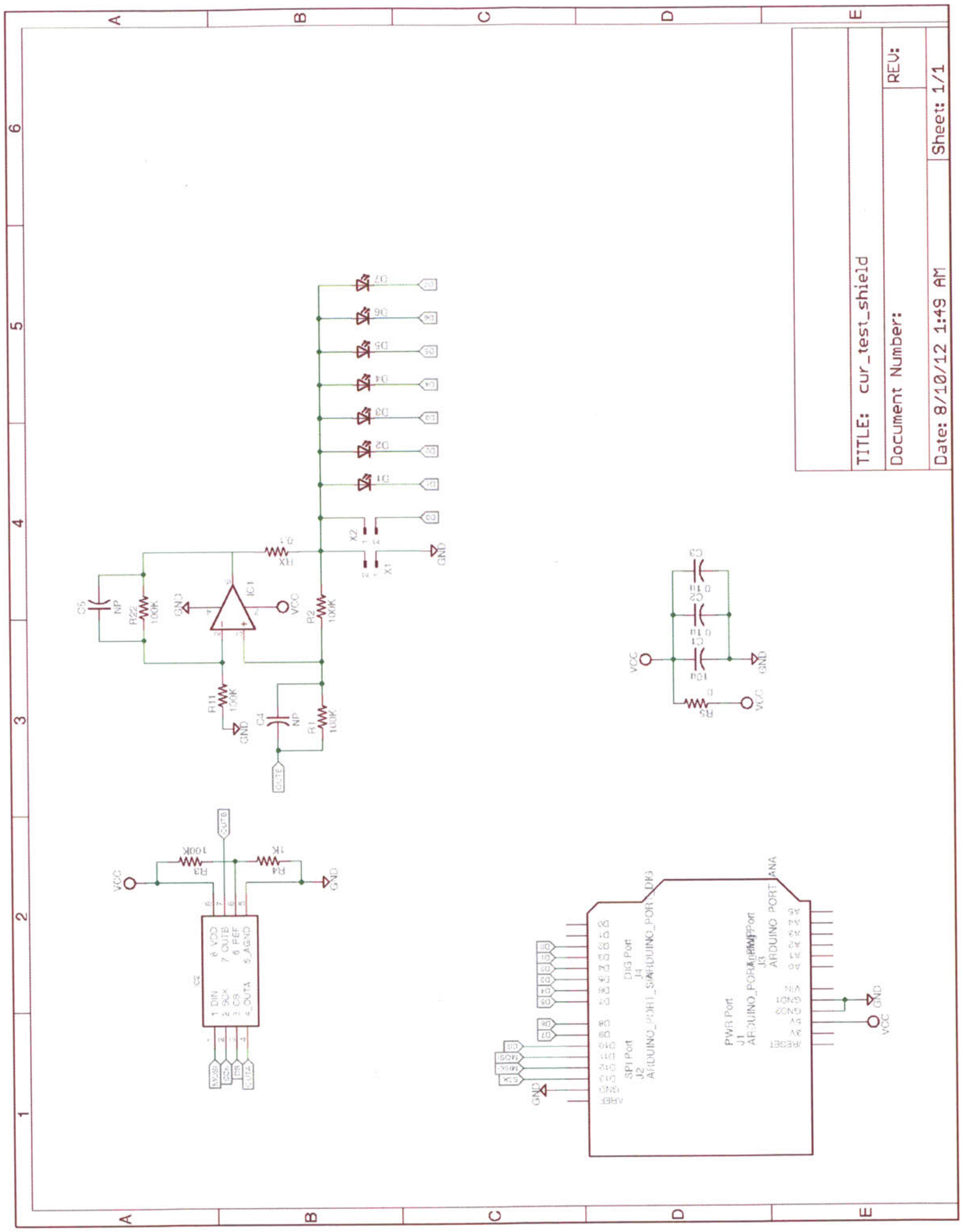


Figure 5-18: Improved Howland current pump schematic.

Despite its advantages and historical context, the Howland current pump was not frequently employed during the course of this thesis. Typically, devices were biased with either positive or negative current sources, and flexibility was not required.

5.4 Driving High-Current Illuminators

Some devices such as high power LED or laser illuminators require very high currents at comparatively low voltages, say 4 V at 20 A. While linear supplies can be designed to operate with these parameters, typically switch-mode constant current supplies are employed. Switch-mode, or switching, power supplies are often more complex than linear supplies, but they are much more efficient, boasting efficiencies upwards of 95%.

5.4.1 Switch-Mode Constant-Current Supplies

Switch mode power supplies have many different topologies, though some common topologies are either "boost," stepping a low voltage up to a higher one, or "buck," dropping a high voltage to a low voltage. Typically, when driving high power semiconductor illumination devices, constant current buck regulators are employed.

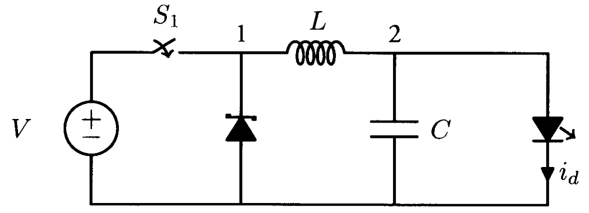


Figure 5-19: Simple constant-current switch-mode buck regulator.

A schematic of a constant-current switch-mode regulator is illustrated in Fig 5-19. In the figure, S_1 is switched rapidly on and off, typically with a carrier from around 100 kHz to 10 MHz, modulated with varying pulse widths (PWM) to give a changing duty-cycle or percent-on-time.

So, if the percent-on-time of the switch is 65%, the time average voltage past the switch at Node 1 is $0.65V$. Now, the waveform past the switch is rather choppy at node 1, so a filter composed of L and C is used to smooth the output of the regulator (Node 2) before the load D.

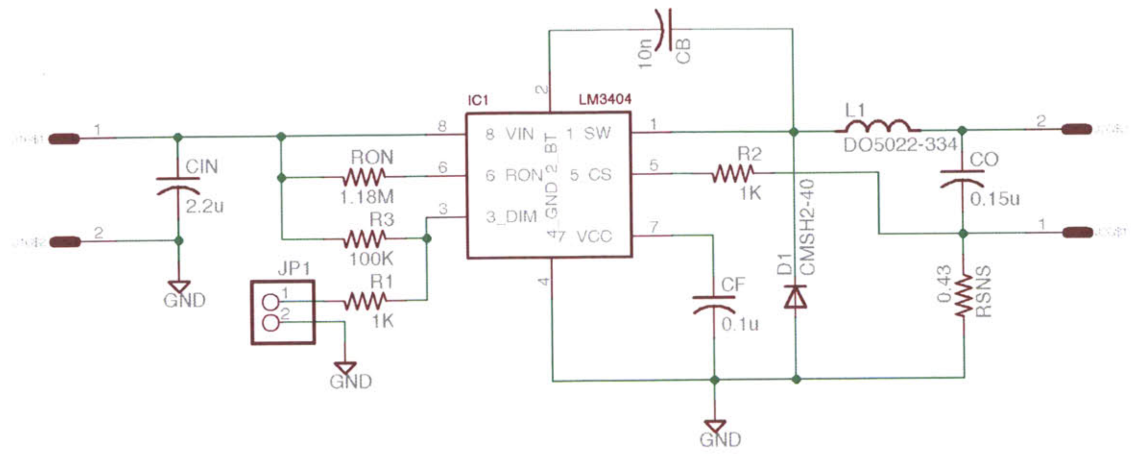


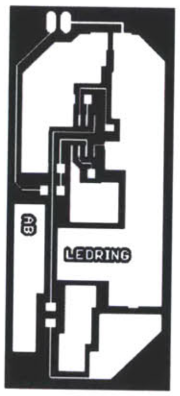
Figure 5-20: Schematic of switch-mode constant-current solid-state illumination controller realized with a LM3404.

In a constant current regulator, the current through the device i_d is monitored and compared to some set point I_d. A controller (not pictured) adjusts the percent-on-time of the switch to such that the current i_d is as close as possible to I_d. In practice, a MOSFET is typically used as a switch and often, a dedicated stand-alone controller is employed. Frequently, the controller and FET are located in the same package. Many manufacturers offer solutions for a wide range of power applications. With the rise in popularity of solid-state lighting, a number of constant-current regulator solutions exist.

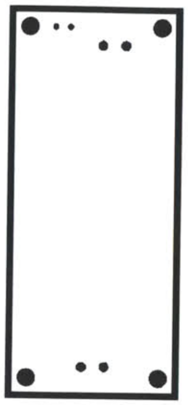
While several switch-mode constant-current power supplies were developed during the course of this thesis, solutions centered around the Texas Instruments (formally National Semiconductor) LM3404 will be discussed [31].

Similar to the regulator illustrated in Fig 5-19, this controller is a buck regulator. With reference to the Fig 5-19, the LM3404 contains the switch S and its respective control circuitry.

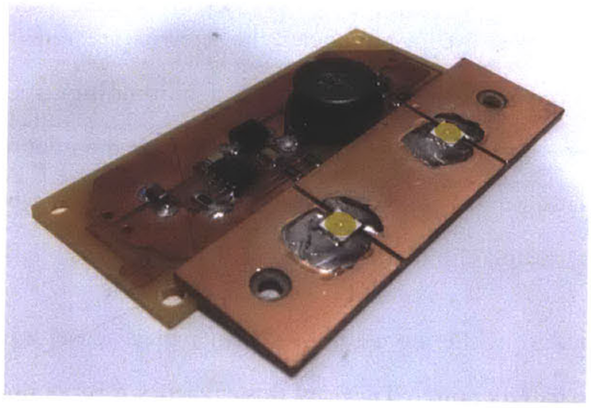
In particular, the LM3404 has an integrated 1.2 A I_{DS} N-channel MOSFET. The control system contained within the LM3404 consists of circuitry to generate waveforms. These circuits have a variable duty-cycle dependent on the voltage sensed through some feedback path or, perhaps, current sensed indirectly through a resistor. With a few external passive components, a constant-current solid-state illumination controller is realizable as shown in Fig 5-20. The controller illustrated in Fig 5-20 is designed to drive a series chain of ten Avago ASMT-J ultra-bright white LEDs with a voltage drop of 2.25 V at around 500 mA [40]. In particular, the components R_{on}, R_{sns} and L_1 were selected to optimally drive the LED string.



(a) Top layer traces and pads.



(b) Holes and board outline.



(c) Fabricated and populated LM3404 constant-current switch-mode power supply PCB with Avago ASMT-J ultra-bright white LED PCB.

Figure 5-21: Constant-current switch-mode power supply diagram of traces and cutout layer, fabricated and populated power supply and LED circuit boards.

The schematic pictured in Fig 5-20 was captured and an electrically consistent printed circuit board was designed and manufactured. The circuit board was designed to follow 10 mil spacing and 10 mil trace width layout rules for a single-sided PCB. All components are surface mount except for the input headers. Once the board was routed, a prototype circuit board was milled out of copper clad FR-1 in-house on a Roland MODELA MDX-20.

Similar to other prototype boards described in this thesis, the circuit was milled in two steps. First, the traces and pads were milled using a 0.4 mm square-shoulder center-cutting endmill, as shown in Fig 5-21a. Next, the board outline, slots, tabs and drills were cut using a 0.8 mm square-shoulder center-cutting endmill, as shown in Fig 5-21b. A photo of the completed circuit board is shown in Fig 5-21c. In the photo, the constant current switch-mode power supply circuit board is pictured with two of the Avago ASMT-J ultra-bright white LEDs mounted to their own carrier PCB.

A side effect of switch-mode power supplies is that the device's intensity may ripple depending on the effectiveness of the output filter. A number of factors work to mitigate this ripple, chiefly the inductor L and capacitor C. In addition, the semiconductor device may exhibit some inertia. The extinction rate of the phosphor in white LEDs will be no quicker than hundreds of nanoseconds. In addition, the extinction rate of the gain medium of a solid-state laser will probably be no quicker than tens of nanoseconds.

In typical spectroscopy topologies, the detector integrates for milliseconds to tens of milliseconds, smoothing any ripple present in the illuminator. In some high-speed spectroscopy applications, this may become an issue. That said, some ripple, or amplitude modulation, in the illuminator is sometimes desired. A modulating illumination source can be used to form a heterodyned detector, increasing the signal-to-noise ratio and eliminating steady-state (DC) errors.

This design, while more complex than typical linear supplies, is very efficient. In some applications, such as battery powered devices, this efficiency is very important. In others, switch-mode power supplies might simply be the most cost effective option. In the case of a high-current linear supply, large, and consequently costly, components must be employed to achieve the desired performance. As discussed in the next section, this is sometimes warranted. Still, switch-mode power supplies dominate commercial and industrial solid state illumination.

5.4.2 High-Current Linear Supplies

There are some cases where it is desirable to drive high-current devices with a linear constant-current supply rather than a switch-mode supply despite its associated drawbacks. In applications requiring very short high-intensity illumination, rapid pulses, or codes, switch-mode constant-current power supplies become problematic.

Switch-mode power-supplies are effectively control systems that are programmed to approach some set current or voltage typically through analog feedback. They have characteristics common with other control systems such as settling time, overshoot, stability, and steady-state errors. If the bandwidth of the controller is low, the settling time will be long. If the supply is pulsed on and off, the settling time is sometimes referred to as rise or fall time.

If the illuminator is flashed quickly or pulsed on and off, the bandwidth of the switch-mode controller is often not large enough to regulate the output in such a short duration. Thus, for high power pulsed illuminator applications, linear supplies are preferred.

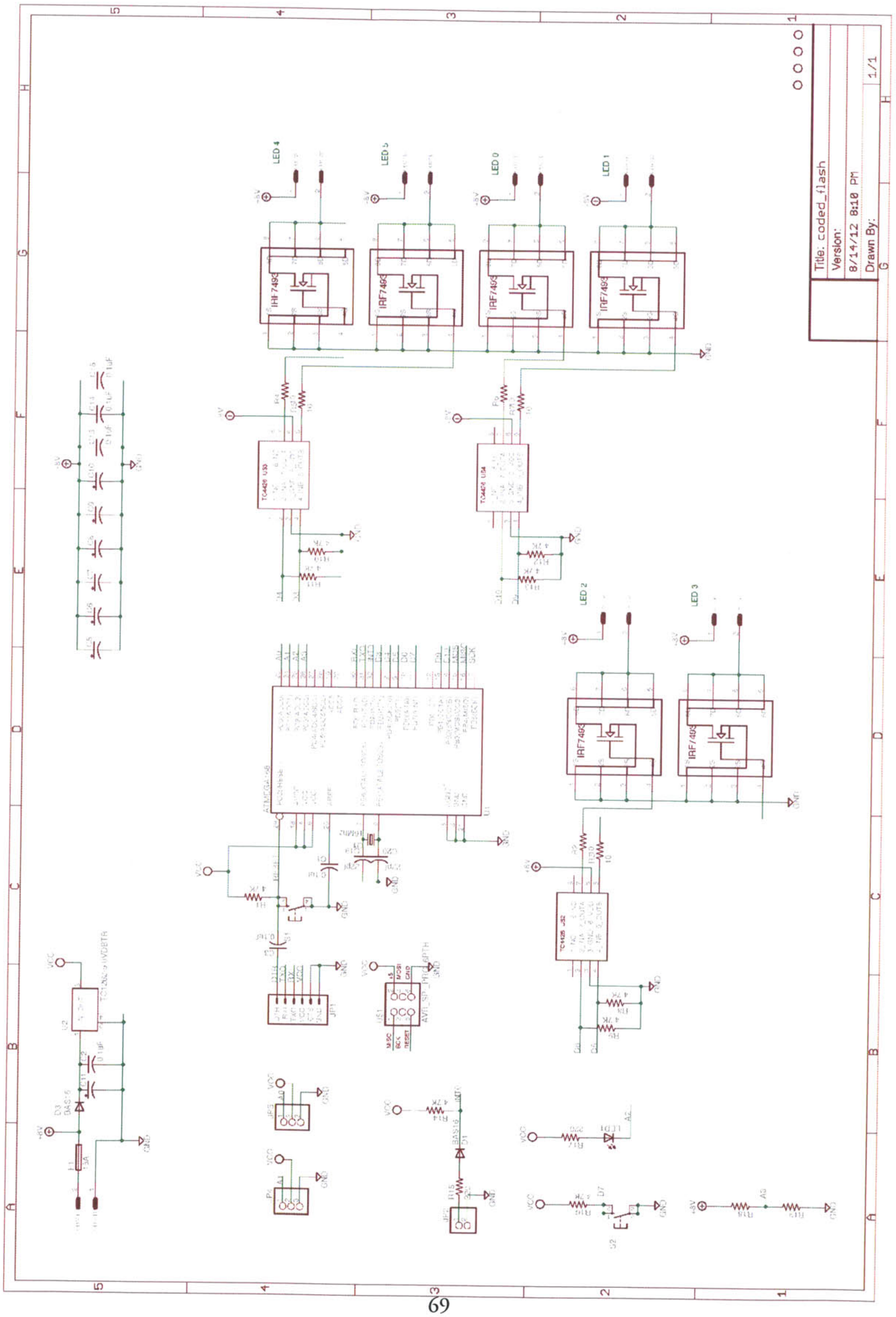
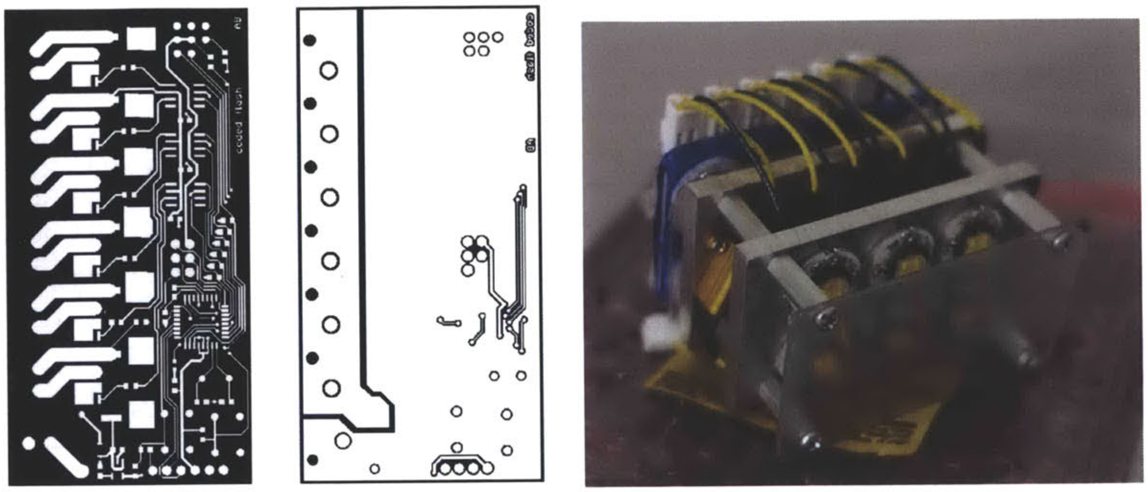


Figure 5-22: High pulse current LED driver schematic.



(a) Top layer artwork. (b) Bottom layer artwork. (c) Fabricated and populated linear power supply with LiPo battery, resistors and SSC P7 LEDs.

Figure 5-23: Constant-current switch-mode power supply diagram of traces and cutout layer, fabricated and populated power supply and LED circuit boards.

In this example, a solid-state alternative to a common xenon camera flash was developed. Xenon flash tubes are quite bright, but are illuminated for a very short duration. A xenon flash in a common point-and-shoot outputs around 1 J of light for around 100 µs or 10 kW [3]. This is a lot of power for such a small device. If the illumination time is relaxed to 100 ms, the power is 10 W. While less impressive, it is much more manageable.

The Soul Semiconductor SSC-P7 ultra-bright 10 W LED can output around 2 W continuously. A group of five such illuminators, pulsed for 100 ms, can approximate the xenon illuminator of a point-and-shoot. Though impressively bright, a hot-shoe mounted speedlight can output around 50 J.

If the LEDs are to be pulsed on for less than 100 ms, use of a switch-mode constant-current supply is dubious. The controller will undoubtably have some control bandwidth which will most likely need to be discovered empirically. However, the bandwidth of the system is limited due to the time constant of the output LC filter. In the circuit described by the schematic in Fig 5-20, the inductor $L1$ is 33 µH and the output capacitor C_O is 150 nF. The rise time of the current in the semiconductor device is on the order of 10 m sec severely limiting this circuit's application to pulsed illuminators.

A linear supply is required to rapidly flash semiconductor devices. In a sense, the circuit becomes an improved version of the first circuit depicted in this section, Fig 5-1. In the design pictured in Fig

5-22, a number of features are considered to reduce signal path impedance to maximize the current rise-time. Starting with the power supply, a high-discharge rate Lithium Polymer battery pack is employed. Similar to an Arduino, an ATMega368 signals Microchip TC4428 1.5A MOSFET drivers to drive 10A N-channel MOSFETs bypassed by 47 μF per channel to further reduce drive impedance. The traces are kept short and large power and ground planes are employed to further reduce inductance. To limit the current flowing through the devices, each LED is connected in series with a 2 Ω 25 W power resistor.

The circuit was routed and the board artwork is shown in Fig 5-23a and 5-23b. This board was manufactured by a third party and designed with the rules for six mil traces, six mil spacing and a minimum drill size of fifteen mils. The power entry connector is in the lower left of the board. Notice the large ground and power planes dominating the bottom layer and top left layer. These planes reduce impedance at the semiconductor devices, increasing current rise-time.

High-current pulsed illumination devices are often used in spectroscopic applications. Here, the system is severely photon-limited but also power-limited; therefore, it is pulsed. Other applications for high-power pulsed illumination are fluorescence and Raman spectroscopy. As ultra high-power illumination devices advance in power efficiency and power output, their application to spectroscopy will only increase. Increased (lumen per watt) may give impetus to novel high-current low voltage pulsed power supply design.

Chapter 6

Illumination Sources

6.1 Introduction

In spectroscopy, two major classes of illuminators are considered: those with broadband, or continuum, output and those with narrowband spectra. Each class of illuminator comes with its own measurement and design philosophy and finds application across a variety of measurement topologies.

6.2 Black Body Radiators

A traditional electro-optical illuminator is the lightbulb, a form of black body radiator. Black body radiation is the electro-magnetic spectrum emitted from an object in thermal equilibrium with it's surroundings. The radiation from tungsten incandescent bulbs closely approximates that of a black body [22].

A tungsten incandescent lightbulb is constructed of an evacuated glass envelope filled with a low pressure mixture of inert gas. A filament is suspended from two electrodes in the envelope. When a current is passed through the filament, it is heated and glows according to a black body of its temperature. The inert gas in the envelope prevents the filament from oxidizing and combusting.

The spectral emission of a black body radiator $B_\lambda(T)$, in terms of the temperature of the body in degrees

Kelvin T, is described by Planck's Law. As described below in equation 6.1 [19], c is the speed of light, h is Planck's constant, and λ is the wavelength.

$$B_\lambda(T) = \frac{2hc^2}{\lambda^5} \frac{1}{e^{hc/(\lambda k_b T)} - 1} \tag{6.1}$$

Typically, incandescent illuminators are isotropic and approximate a Lambertian or near-Lambertian (diffuse) source. Black body illuminators play an important, though limited, role in modern spectroscopy. They are very broadband, more so than any other illumination source. In some applications this is necessary; in others, however, they are *too* broadband - most of their light output is discarded because it is outside the sensitivity range of the detectors. Their ability to convert energy from electrical to optical is highly efficient, but, their application becomes less attractive if a majority of their light output is discarded.

Another problem is that incandescent bulbs are intrinsically less robust than their solid state counterparts. They are fragile and can easily fail if subjected to physical shock or vibration. In addition, the bulbs wear as the tungsten evaporates, resulting in a significantly shorter lifetime than their solid state counterparts.

6.3 Arc and Flash Lamps

An arc lamp is constructed of two electrodes in a glass envelope filled with a low pressure gas. A high voltage strike is used to form a plasma in the globe, which emits optical radiation. Flash lamps are similar, except they are used in pulsed rather than continuous operation. As such, arc and flash lamp electrodes are shaped differently to support either pulsed or continuous operation [22].

Arc and flash lamps are thermal devices, but they operate in the unstable rather than stable regime, so their output does not represent black body radiation. Their spectra are typically dominated by strong spectral lines corresponding to electronic transitions in the fill gases with a weaker, background black body.

In spectroscopy, arc and flash lamps are often employed as bright ultraviolet sources. Typically, lamps designed to radiate in the ultraviolet are filled with mercury vapors or deuterium (^2H or heavy hydrogen).

With the exception of nitrogen lasers, arc and flash lamps are the only way to produce high output ultraviolet radiation.

Arc and flash lamps are seldom employed in spectroscopic applications outside the ultraviolet due to their uneven spectroscopic characteristics, high optical and electrical noise, and complex electrical requirements. Though more robust than incandescent light bulbs, arc and flash lamps are also mechanically fragile and sensitive to shocks and vibration. They also can be physically oversized for the amount of light energy they produce to limit high voltage flashover.

6.4 Discharge Tubes

Low-pressure gas discharge tubes are constructed much like arc and flash lamps. A glass envelope is filled with low pressure gas which is then electrically excited via two electrodes. As the gas decays to its ground state, photons are emitted. These emitted photons correspond to the energy difference between the excited and ground states, producing fairly narrow spectral lines dependent on the composition of the gas in the tube. Low-pressure gas discharge tubes are useful for producing moderately monochromatic light. This light can be tuned depending on the gas composition. A common example of a low-pressure gas discharge tube are neon signs and lighting.

6.5 Light Emitting Diodes

Light emitting diodes (LEDs) are convenient illumination sources for spectroscopy applications and are used extensively throughout this work. Light emitting diodes are formed by doping a semiconductor with impurities to produce a PN junction. The band gap of the semiconductor junction is carefully designed such that when the device is forward biased, light of a particular wavelength is emitted. When a positive current is applied, electrons flow from the cathode to the anode and their corresponding holes flow from the anode to the cathode. When an electron meets a hole as the charge carriers travel through the device, it drops into a lower energy level, releasing energy in the form of a photon. Energy not emitted as photons is lost as heat through lattice vibrations [20].

LEDs boast a number of advantages over other illuminators which make them particularly suited to spectroscopy and simple to develop devices and instruments around. By comparison to other illumination sources, they have simple electrical drive requirements, requiring low voltages and modest currents to illuminate. In addition, they are small, physically robust, and they have long lifetimes. LEDs can be modulated electrically to transmit data or for use in heterodyned measurements. They are efficient, converting much of the applied current to photons. Since they are relatively narrowband, much of that energy can be practically used in the optical system. Light emitting diodes are discussed and tested extensively in Chapter 7.

6.6 Lasers

In a laser, a gain medium in an optical resonator is pumped electrically, optically, chemically or thermally to achieve stimulated emission and subsequently, population inversion. Because of their construction, the resonator and gain medium have a small support, giving lasers very high spatial and temporal coherence. Since laser have very high temporal coherence, they have very narrow spectral bandwidth [24].

A variety of materials are employed as laser gain media, each with varying spectral output characteristics. Laser classes are often determined by the type of gain media employed.

- Solid State - ND:YAG, Ti:Sapphire, ND:Glass
- Gas - CO_2, Argon, Nitrogen
- Dye - Rhodamine, Fluorescein, Quinine
- Diode - GaAs, InP, GaN
- Fiber - Er:fiber, Yr:Fiber

In spectroscopic applications, diode, solid state, and dye lasers are used most often. Recently, however, fiber lasers are becoming popular. Diode lasers operate similar to light emitting diodes but form an optical resonator on their die. This gives diode lasers far higher spatial and temporal coherence than light emitting diodes. Development of diode lasers has benefitted significantly from consumer electronics development. In particular, diode lasers are employed widely in CD and DVD players, commoditizing their development.

Solid state lasers, one of the first types of laser, are often employed in sophisticated spectroscopy systems that pulse rather than operate continuously. When a solid state laser is pulsed sufficiently quickly (e.g., by Kerr lens mode locking), the temporal coherence is necessarily short increasing the bandwidth of the laser. Thus, a broadband, bright pulsed source is formed [48].

Dye lasers are rather interesting for a number of reasons. For one, the emission wavelength can be easily tuned by altering the dye. There are a wide variety of dyes which can be coaxed into lasing. Unfortunately dye lasers are rapidly falling out of favor for a number of reasons. The dyes used in dye lasers are often rather toxic, carcinogenic and corrosive. Also, the laser dyes bleach and, after a number of uses, will no longer lase. That said, dye lasers still find application because they have very high gain and can be tuned with high selectivity.

Chapter 7

Illuminator Characterization

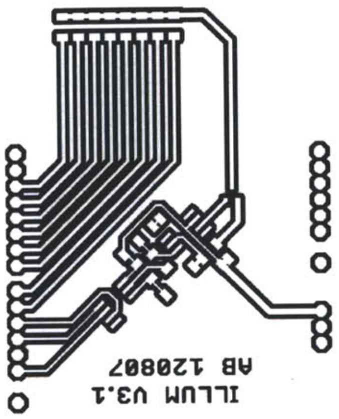
7.1 Introduction

One of the first steps in constructing a sensor is selecting illuminators, often by techniques described in Appendix A). Before the sensor is completed, the illuminators are further characterized to ensure consistent operation with their specifications or, more commonly, augment insufficient manufacturer documentation. This section details procedures used to characterize the illuminators employed in the sensors constructed during the course of this thesis. These procedures include determination of the device electro-optical characteristics, particularly their conversion efficiency and spectral characteristics.

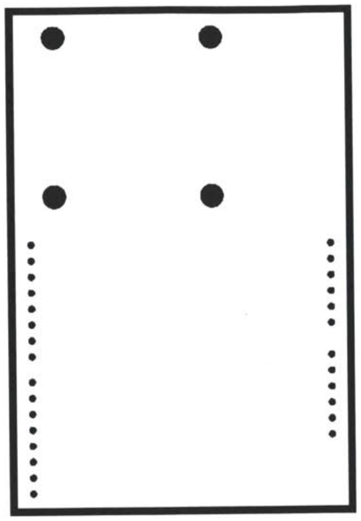
7.2 Multi-Device Text Fixture

To characterize common illuminators employed during the course of this thesis, several test fixtures were prototyped. The fixture discussed was developed to evaluate light emitting diodes in surface mount 1206 packages. The fixture, based on designs discussed in Chapter 5, can source variable amounts of current from 0 to 40 mA for up to eight devices. The fixture is built as an Arduino shield to be carried atop the development platform discussed in Appendix C, allowing straightforward development and testing.

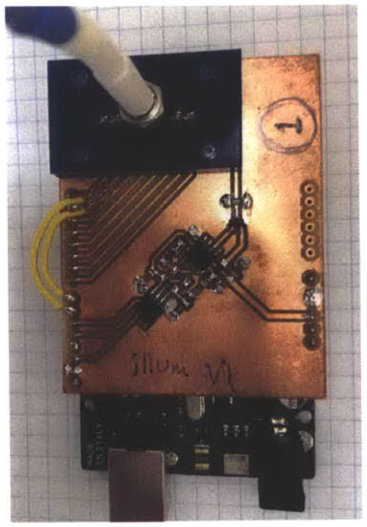
The constant current supply is a low-side linear, supply consisting of a serially-attached digital-to-analog converter driving an op-amp and a current pass N-channel MOSFET. The design is electrically similar



(a) Top layer traces and pads.



(b) Holes and board outline.



(c) Early version of fabricated and populated fixture. Note the bodge wires.

Figure 7-1: Diagram of traces and cutout layer, fabricated and populated circuit board. The blue fixture allows fiber connection for the spectrometer.

to that pictured in Fig 5-5. The design was extended slightly, and modified to suit components on hand. The full schematic of the test fixture is shown in Fig 7-2. The design was manufactured in-house, similar to the other prototypes discussed in this thesis. The design was milled on a Roland Modela using a 40 mm endmill. This requires the layout to follow 10/10 mil trace width/spacing conventions. To that end, the smallest trace width in the design is 16 mils, all LEDs and passive parts are 1206 and the integrated circuits are all SOIC8. The N-channel MOSFET is in a SOT-23 package.

For this evaluation, a set of LEDs were selected from the database described in Appendix A. Those LEDs were selected because the fixture used here is built to accept LEDs in 1206 packages and the fixture can sink 40 mA at most. The LEDs chosen have maximum wavelength variability, which will increase the diversity of potential measurements. These LEDs and their published center wavelength are listed in Table 7.1. In total, sixteen LEDs were selected for this evaluation.

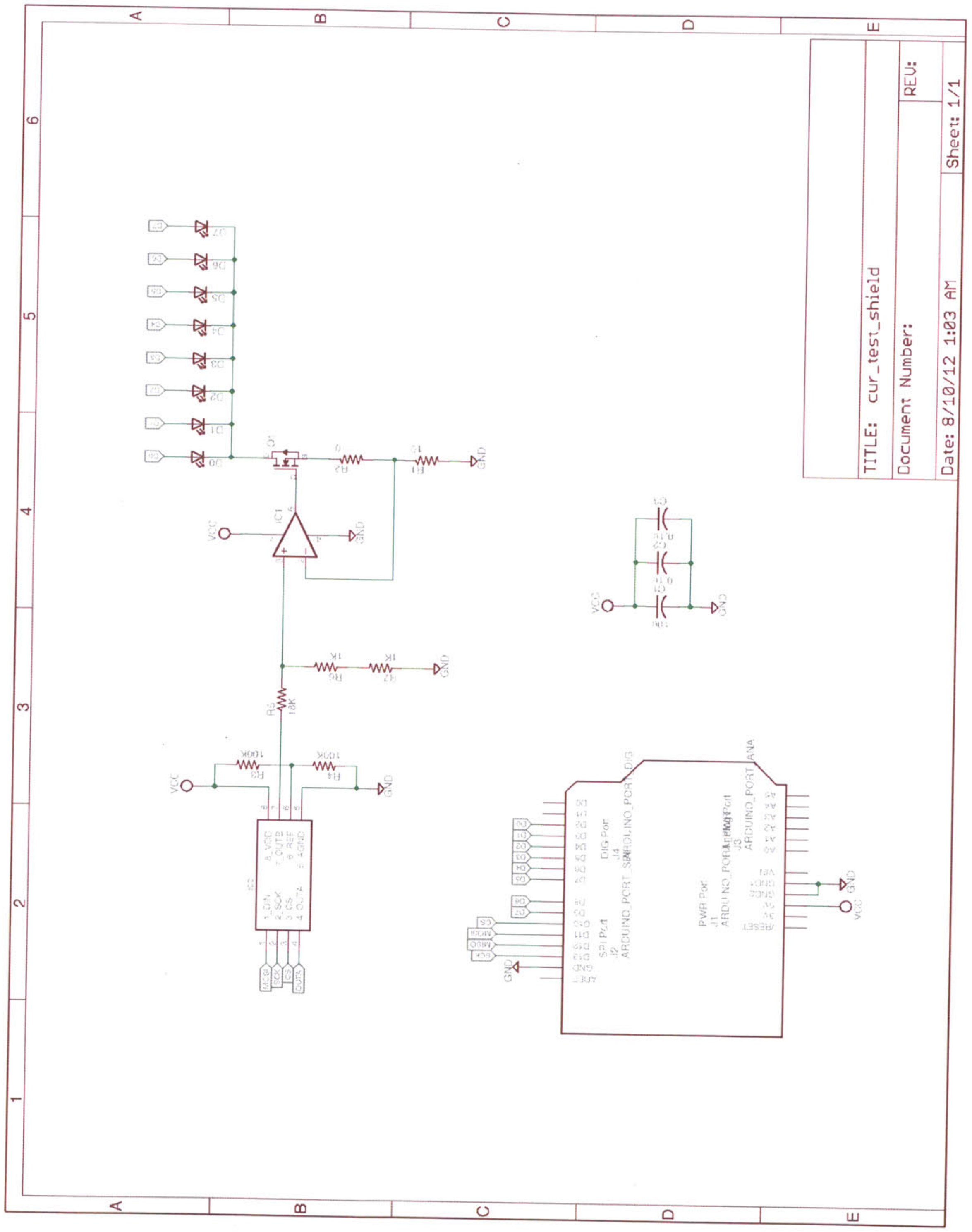


Figure 7-2: Schematic of multi-device test fixture.

Manufacturer	Part Number	λ_{center}	Fixture	Pin
Chicago Miniature Light	CMD15-21UBC/TR8	430 nm	1	D2
Kingbright Corp	APT3216QBC/D	470 nm	1	D3
Kingbright Corp	APT3216CGCK	570 nm	1	D4
Lite-On Inc	LTST-C150KGKT	571 nm	1	D5
Lumex Opto Inc	SML-LX1206SYC-TR	590 nm	1	D6
Lite-On Inc	LTST-C150KSKT	591 nm	1	D7
Lumex Opto Inc	SML-LX1206SOC-TR	610 nm	1	B0
Everlight Electronics	QTLP650CRTR	624 nm	1	B1
Kingbright Corp	APTD3216EC	625 nm	2	D2
Lite-On Inc	LTST-C150KRKT	631 nm	2	D3
Lumex Opto Inc	SML-LX1206SIC-TR	636 nm	2	D4
Panasonic	LNJ218C82RA	645 nm	2	D5
Stanley Electric Co	HBR1105W-TR	647 nm	2	D6
Chicago Miniature Light	CMD11-21SRC/TR8	660 nm	2	D7
Osram Opto Semi	SFH 4059-Z	860 nm	2	B0
Everlight Electronics	IR26-21C/L110/TR8	940 nm	2	B1

Table 7.1: LEDs characterized using the test fixture.

Examining Table 7.1, there is good wavelength coverage from 430 nm ultra-violet to 940 nm near-infrared illuminators. Though some solid-state illuminators have center wavelengths further into the ultra-violet and infrared, they were not evaluated with this fixture due to their extreme cost, unusual mechanical packaging, or incompatible electrical requirements. Most importantly, the spectrometer used in this study is sensitive to wavelengths from 350 nm to 1000 nm, disqualifying wavelengths outside of its range.

The firmware developed for the constant current supply described in Fig 5-5 was employed for this test fixture. To ascertain appropriate device currents, the GUI designed for the circuit in Fig 5-5 was also used.

7.3 Characterization Methodology

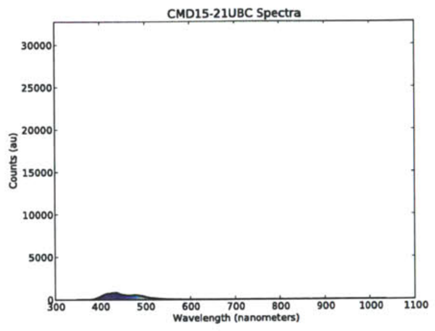
Once constructed, the group of sixteen LEDs in Table 7.1 were ordered according to wavelength from 430 nm to 940 nm. The LEDs were then split into two groups. The first group has the eight LEDs ranging from 430 nm to 624 nm. The second group has the eight LEDs ranging from 625 nm to 940 nm. Each group was attached, in order by wavelength, to a test fixture. The LEDs were soldered to the fixture in reverse order with respect to their wavelength. To accommodate all sixteen devices, there were two fixtures used. "Pin" is the pin on the ATMega368 controller on the Anduino (Appendix C) development board. For example, a Pin of B0 corresponds to Port B, pin 0 on the ATMega368. "Number" refers to the LED number called out in the firmware, software, and GUI.

To measure the spectral characteristics of each illuminator, an Ocean Optics USB4000 VIS-NIR fiber attached USB spectrometer was employed [34]. The USB 4000 is a handy device, it is a rather compact, simple to use cross-platform spectrometer. The SMA fiber needs only to be connected and using some host software, the spectrum of the incoming light can be measured. The spectrometer is sensitive from approximately 350 nm to 1000 nm. The spectrum falls on a 3648 element linear CCD, giving the spectrometer approximately a 0.18 nm resolution. The CCD is digitized with a sixteen analog-to-digital converter. The entrance aperture of the spectrometer is given by the aperture of the fiber used to couple light into the device. In the following experiments, a 600 μm diameter fiber was employed.

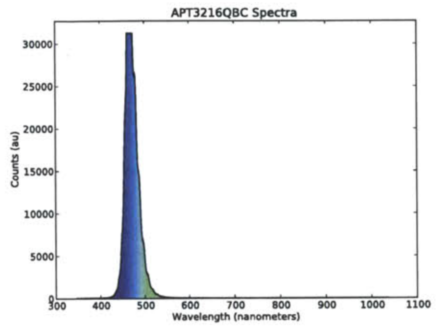
To facilitate LED characterization, a small Python application (Appendix F) was authored. The application is controlled via a primitive command line user interface. First, the application establishes communication with the Ocean Optics USB4000 spectrometer and the test fixture. Once both the fixture and the spectrometer are successfully communicating with the host PC, the calibration parameters are read from the spectrometer and a dark, or background, spectrum is captured. The background spectrum is used to subtract any ambient light contaminants which may corrupt the measurements. Despite this dark frame, the measurements were performed in a light-tight enclosure to minimize stray light.

The user is then prompted via the command line to select which device to illuminate and the filename to save the result figure and data. An additional dark frame is taken. The selected LED is illuminated and the spectrometer takes five readings and averages them internally. The dark readings are subtracted from the averaged illuminated readings. The resulting spectrum is plotted and saved as comma separated values for future analysis. Results of the analysis are presented below in Fig 7-3.

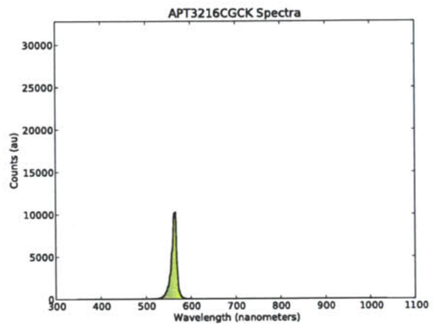
To objectively measure the electro-optical properties of the devices, the current through each device was identical. By inspection, a DAC code C 100 of 12 bits or, by equation 5.10, 1.2 mA was chosen for this test.



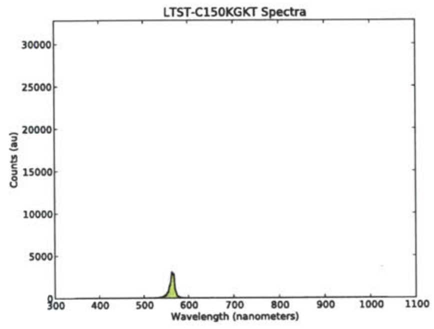
(a) CMD15-21UBC.



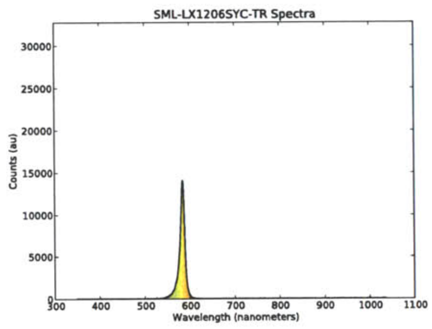
(b) APT3216QBC.



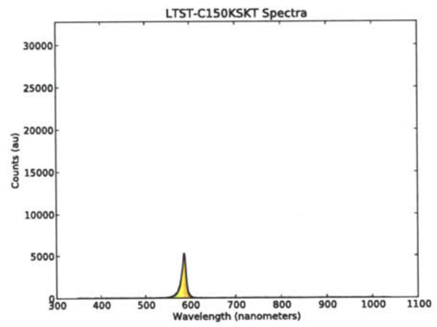
(c) APT3216CGCK.



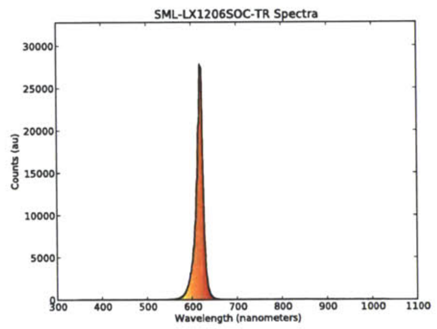
(d) LTST-C150KGKT.



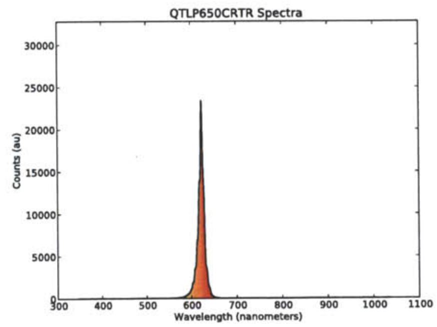
(e) SML-LX1206SYC-TR.



(f) LTST-C150KSKT.



(g) SML-LX1206SOC-TR.



(h) QTLP650CRTR.

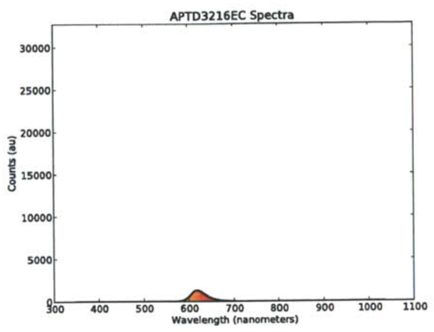
Figure 7-3: Fixture 1 LED spectra. LEDs range from 420 nm to 624 nm center wavelength.

In Fig 7-3, the spectra of the eight LEDs on test fixture one are illustrated. These LEDs range from a nominal center wavelength of 420 nm to 624 nm. The recorded spectra show a noticeable variance in conversion efficiency. For a given device current i_d, some devices produced significantly more photons than others. In particular, the blue 470 nm APT3216QBC LED and the red 610 nm SML-LX1206SOC-TR LED were particularly bright while the blue 420 nm CMD15-21UBC LED was particularly dim.

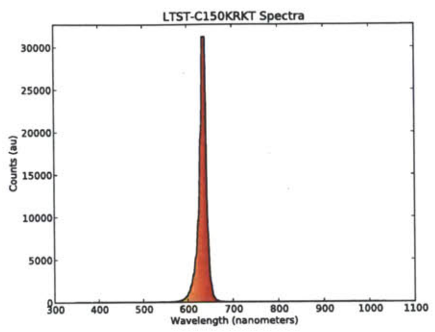
It is important to note that despite the spectrometer's published range from 350 nm to 1000 nm, external optical components (e.g., fibers, collimation optics) do not have guaranteed bandwidths. In particular, the fiber used to guide the light from the illuminators to the detector, is specified from 400 μm to 2.2 μm. Though the LEDs are expected to fall within the system (fiber and spectrometer) bandwidth, the manufacturer warns that towards the edges of the rated bandwidth of the fiber, attenuation may occur. From cursory human inspection, it is apparent the fiber attenuates the CMD15-21UBC 420 nm LED significantly.

The spectra of the eight LEDs on test fixture two appear in Fig 7-4. Many of these LEDs are more light efficient than their blue and green counterparts on test fixture one. This is, in part, intrinsic; red and near-infrared LEDs are typically more efficient than their counterparts due to the band-gap of silicon. While blue LEDs are attenuated, infrared LEDs fall in the "sweet-spot" of the detector and may appear brighter as a result. In fact, silicon's band-gap, the same property that makes red and near infrared LEDs more efficient, also increases the detector's sensitivity to those wavelengths which may further increase their apparent brightness. While silicon sensitivity peaks at around 825 nm, its sensitivity falls off rapidly. Furthermore, depending on the spectrometer design, vignetting may attenuate the spectrometer's response at the edges of the wavelength sensitivity.

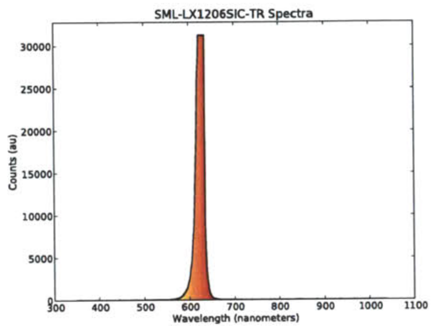
It is difficult to compensate and calibrate for non-linearities in any spectroscopic system. This is largely because it is difficult to construct a truly white illumination source. Every illumination source suffers from resonant modes which produce strong absorption and transmission lines. Even the simplest idealized illumination source, the black body, cannot be white. That said, one could imagine schemes for producing white illuminators and calibration processes; unfortunately, those are beyond the scope of this thesis. Ocean Optics provides rudimentary calibration coefficients which were used when gathering these LED spectra with some positive effect.



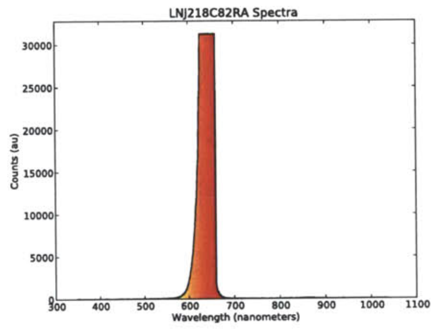
(a) APTD3216EC.



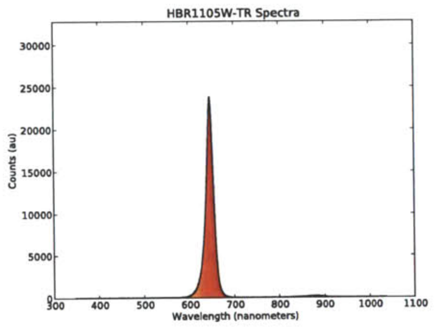
(b) LTST-C150KRKT.



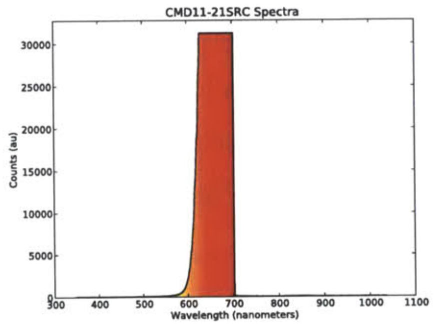
(c) SML-LX1206SIC-TR.



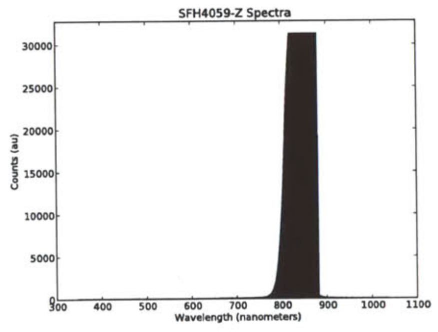
(d) LNJ218C82RA.



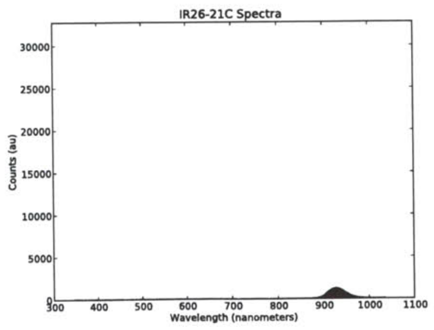
(e) HBR1105W-TR.



(f) CMD11-21SRC.



(g) SFH 4059-Z.



(h) IR26-21C.

Figure 7-4: Fixture 2 LED spectra. LEDs range from 625 nm to 940 nm center wavelength.

7.4 Normalized Spectral Measurements

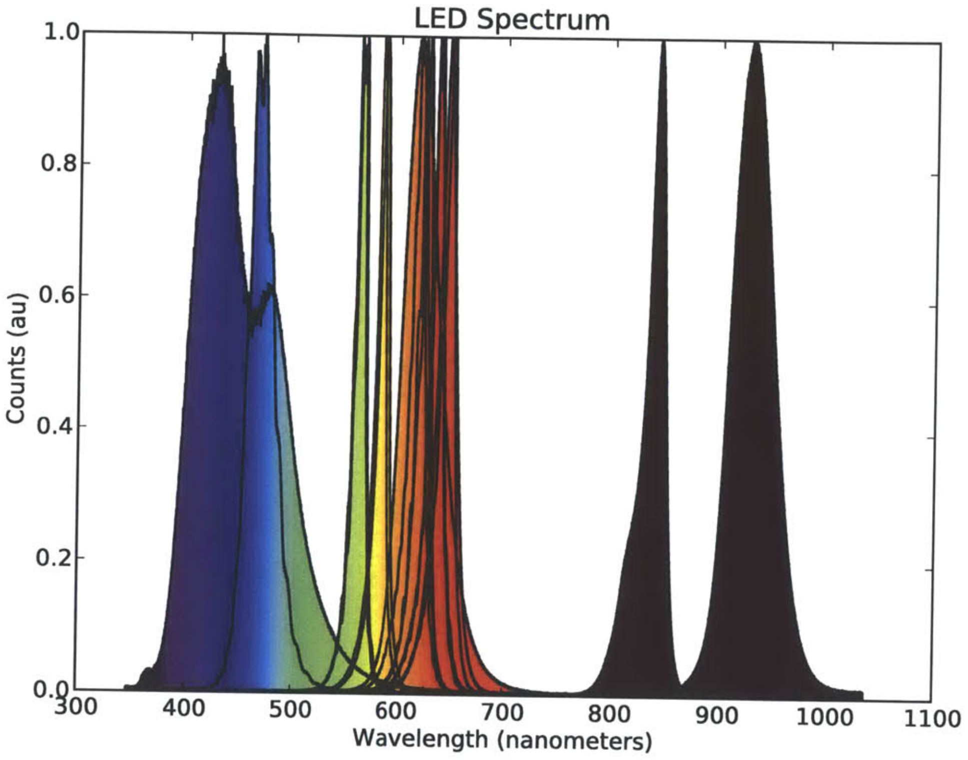


Figure 7-5: LED spectra normalized and plotted together.

To compare the spectrum of each respective LED, software, reproduced in Appendix E, was developed to search through the device drive currents in order to find a current which roughly normalizes the output of each LED. The program searches coarsely through drive currents until approximately equal output flux is achieved. Then, the spectra are scaled in software to have the same range. They are then plotted together on the same figure, as shown in Fig 7-5. In addition, a table (Table 7.2) was constructed the enumerate the discovered device drive currents which roughly normalize their output current. In this table, both the DAC code and its corresponding value in mA are listed.

It's interesting to note clumping of wavelengths in the visible wavelengths. Those LEDs in visible light wavelengths are not degenerate, however, and offer finer discrimination for some spectral features. Visible light LEDs are typically also more spectrally pure than infrared, ultraviolet, and deep blue LEDs. This could be a result of their intended application - there is little impetus to develop spectrally pure

Manufacturer	Part Number	λ_{center}	DAC Code	Current [mA]
Chicago Miniature Light	CMD15-21UBC/TR8	430 nm	4064	49.61
Kingbright Corp	APT3216QBC/D	470 nm	64	0.78
Kingbright Corp	APT3216CGCK	570 nm	192	2.34
Lite-On Inc	LTST-C150KGKT	571 nm	512	6.25
Lumex Opto Inc	SML-LX1206SYC-TR	590 nm	160	1.95
Lite-On Inc	LTST-C150KSKT	591 nm	368	4.49
Lumex Opto Inc	SML-LX1206SOC-TR	610 nm	80	0.98
Everlight Electronics	QTLP650CRTR	624 nm	48	0.59
Kingbright Corp	APTD3216EC	625 nm	768	9.34
Lite-On Inc	LTST-C150KRKT	631 nm	80	0.98
Lumex Opto Inc	SML-LX1206SIC-TR	636 nm	64	0.78
Panasonic	LNJ218C82RA	645 nm	48	0.59
Stanley Electric Co	HBR1105W-TR	647 nm	128	1.56
Chicago Miniature Light	CMD11-21SRC/TR8	660 nm	16	0.19
Osram Opto Semi	SFH 4059-Z	860 nm	32	0.39
Everlight Electronics	IR26-21C/L110/TR8	940 nm	1360	16.60

Table 7.2: Device currents which produce approximately equivalent light output.

LEDs because humans observe them, while infrared and ultraviolet LEDs are invisible to our eyes.

The bluest device, the CMD15-21UBC 420 nm LED, has the largest bandwidth in this set. At such short wavelengths, this bandwidth could arise from fluorescence, a property exploited to produce white LEDs. Another explanation for the bandwidth of this device is the absorbance of shorter wavelengths. As shorter wavelengths are absorbed, the apparent brightness of longer wavelengths increases, making the device appear to have a higher bandwidth.

7.5 Center and Peak Wavelengths

It's interesting to note that LED manufacturers and distributors (particularly Digikey) specify both a device center wavelength and a peak wavelength. In devices which do not exhibit symmetrical spectra, the peak wavelength differs from the center wavelength. In particular, the center wavelength is thought of as a sort of spectral "center-of-mass", while the peak wavelength is considered the brightest spectral frequency.

Using the SFH 4049-Z infrared LED as an example, note that the center and peak wavelengths dif-

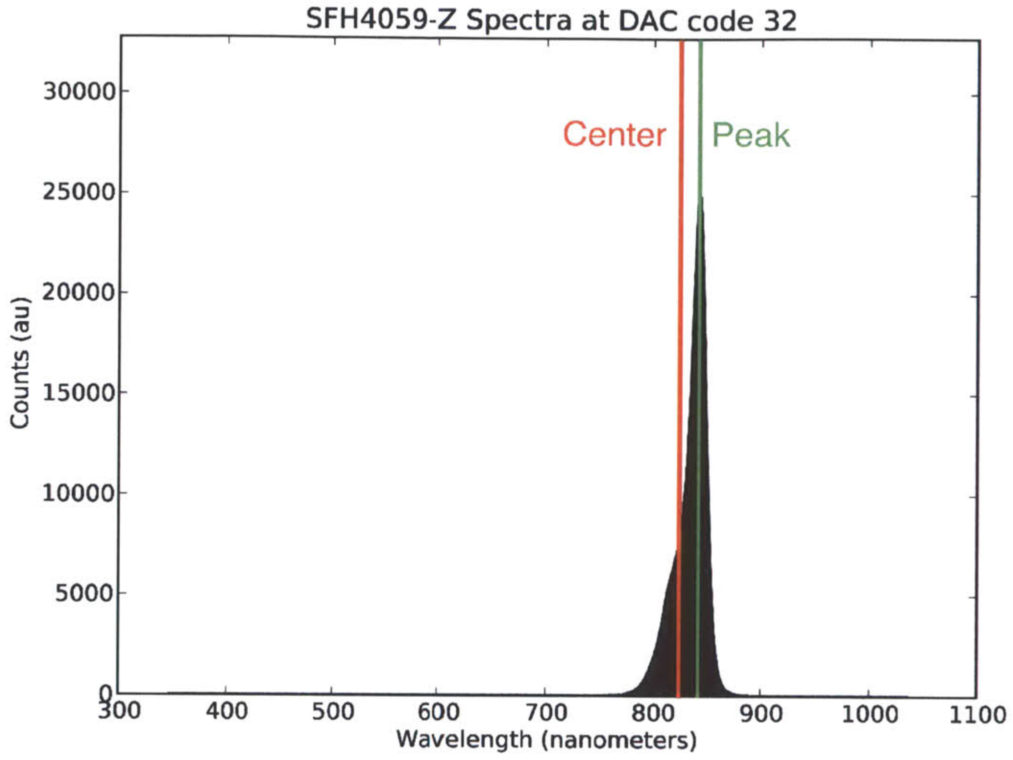


Figure 7-6: Difference between center and peak wavelengths in the SFH 4059-Z. Osram Opto Semi specifies a peak wavelength of 860 nm and a center wavelength of 850 nm.

fer. The spectrum, illustrated in Fig 7-6, is asymmetric. Thus, the peak wavelength is longer than the center wavelength. To illustrate this, in the figure, the center wavelength is drawn in red, and the peak wavelength in drawn in green.

7.6 Discussion

Characterizing illuminators may, at first seem tangential to the core of this work. In fact, it is most central. As the designer constructs their analysis toolbox, the devices are of no use if their characteristics are unknown. Even in the rare case where manufacturers provide comprehensive device documentation, many quoted facts and figures cannot be compared to other devices due to various manufacturer's different and undocumented test procedures. The methodology and results discussed in this section serve to level the discussion of device characteristics. This is crucial when designing spectral sensors.

Chapter 8

Design & Prototypes

8.1 Introduction

In this chapter, the building blocks discussed throughout this thesis are combined to form several working examples of spectroscopic sensors. In general, two approaches are undertaken. In one strategy, supervised transmission spectroscopy, the spectrum of the materials to be discriminated or identified are known. In the other, unsupervised transmission spectroscopy, the spectrum of the samples are unknown. When the samples are well characterized, a device can be constructed which exploits features in the spectrum. However, where the spectrum is unknown, the spectral characteristics are learned. Many of the cases explored lack complete spectral information, which must be subsequently discovered.

8.2 Unsupervised Transmission Spectroscopy

In the case of unsupervised transmission spectroscopy, the instrument learns the spectral response of a sample such that it can be later identified. To do this, a series of general purpose sensors were constructed which identify the pertinent spectral bases of the sample.

Each sensor, pictured in Fig 8-1, is constructed of an Arduino (or equivalent), an illuminator circuit board, a detector circuit board, and a mechanical fixture to hold the optical system and sample in align-

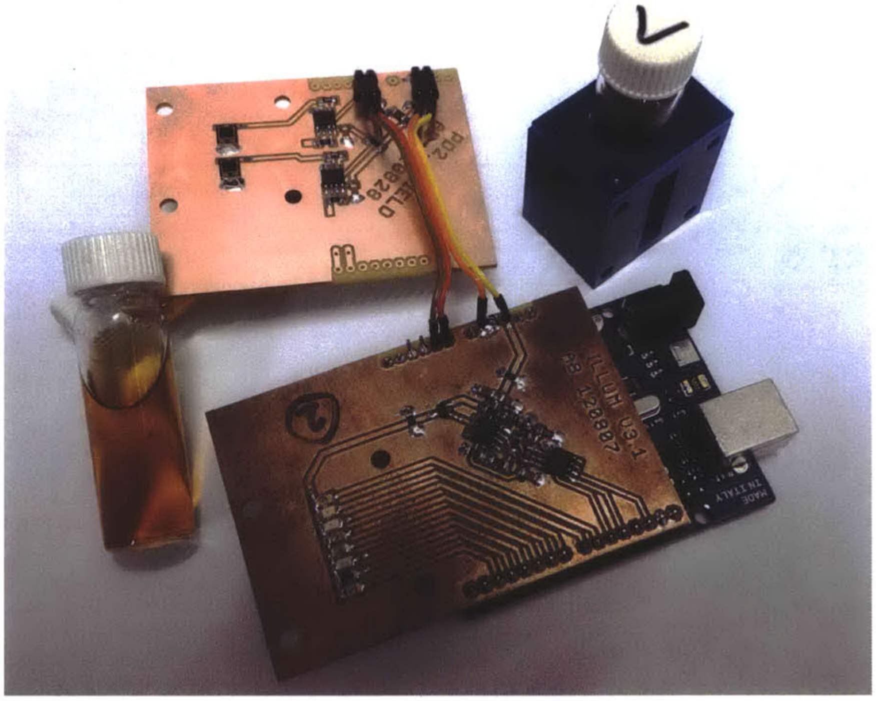


Figure 8-1: Transmission spectrometer disassembled. The programmable illuminator circuit board (bottom) is mated to an Arduino (blue PCB). The detector circuit board is connected via multicolored cable to the illuminator circuit board. The mechanical fixture, which co-aligns the illuminator, sample, and detector, is pictured in the upper right. Also pictured are several samples in borosilicate glass vials.

ment. Both the illuminator and detector circuit boards can be configured to mount directly to an Arduino as a shield. In typical operation, the illuminator circuit board is mated directly with an Arduino, and the detector circuit board is connected via short cables to the illuminator circuit board.

The illuminator circuit board is constructed similar to the fixture discussed in Chapter 7 and Appendix F. Each illuminator circuit board has positions to carry up to eight 1206 package light emitting diodes. The circuit produces a programmable current which can be directed through any of the LEDs to produce a controlled light output. In the circuit, a twelve-bit digital-to-analog converter produces a voltage. The voltage is than converted to a current using a MOSFET buffered operational transconductance amplifier. This and other strategies for producing controlled currents are discussed in Chapter 5 and Appendix E. The complete schematic for the illuminator circuit board can be found in Appendix G.

The detector circuit board is composed of two general purpose photodiode circuits similar to those discussed in Chapter 3. The circuit board carries two TEMD5080X01 photodiodes, each connected in photovoltaic mode to an op-amp transimpedance amplifier. The output of each transimpedance amplifier is buffered with a unity gain non-inverting amplifier and connected to two channels of the microcontroller's 10-bit analog-to-digital converter. Detector taxonomy, and the selection of the TEMD5080X01, is discussed extensively in Chapter 3. Photodiode amplifiers and their modes of operation are explored in Chapter 4. The complete schematic for the detector circuit board can be found in Appendix G.

The illuminator, detector, and Anduino circuit boards were constructed in house. These circuit boards were constructed from single-sided copper-clad FR1 laminate. A Roland Modela MDX-20 desktop milling system fitted with a 0.39 mm square-shouldered center-cutting two flute endmill was used to remove unwanted copper. These circuit boards were all designed to have no smaller than ten mil traces with a minimum of ten mil spacing between traces. A 0.80 mm square-shouldered center-cutting two flute endmill was used to cut out the board outline and cut holes for through hole components and optical fixtures.

The mechanical fixture pictured in Fig 8-1 is constructed of a laminated stack of laser-cut acrylic parts. This fixture holds the detectors, illuminators and a sample vial in alignment. The pictured setup is configured to hold samples contained in 10 mL vials. Fixtures for 5 mL vials, 20 mL vials, and square polypropylene or quartz cuvettes were also developed.

8.2.1 System Linearity

To evaluate the design pictured in Fig 8-1, a fixture was constructed to optically couple the illuminators to the detectors without any sample present. This fixture was constructed of laminated laser-cut acrylic, similar to the fixtures which hold sample vials. To determine the linearity of the system composed of photodetectors and illuminators, the current through each illumination device was varied and the photodetector response was recorded.

To measure the response of the photodiode and LED fixture, software was authored which reads from a csv file containing the system configuration. The system configuration csv file enumerates the manufacturer, model number, and center wavelength λ_{center} of each illuminator soldered to the fixture. In addition, the configuration file enumerates the photodetector configuration on the detector circuit board.

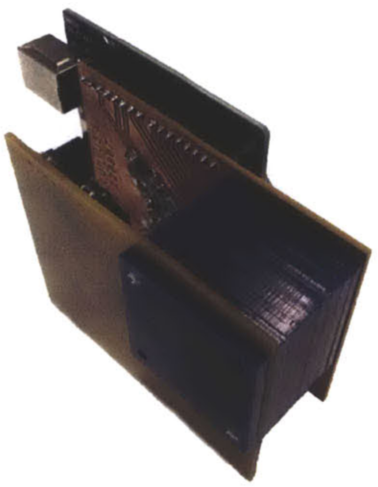


Figure 8-2: Transmission spectrometer setup for characterization and testing.

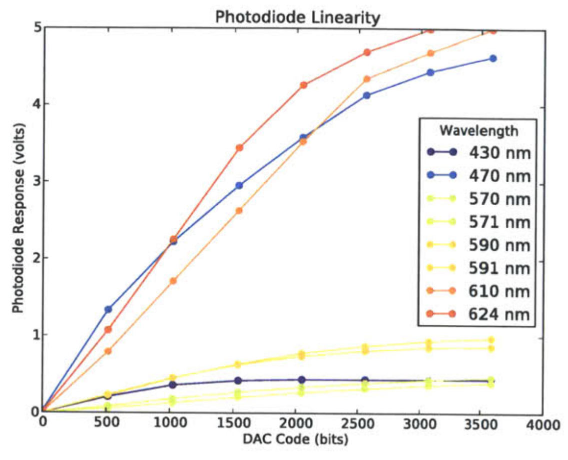
The software, described in Appendix G, first prompts the user to input the serial port the hardware is attached to. Next, the user is prompted to enter the name of the hardware description `csv` file. Lastly, the software scans through a set of increasing LED currents for each LED indicated in the configuration file. The results are plotted below in Fig 8-3 using false colors to indicate the wavelength of each illuminator.

The linearity of the illuminator and photodiode system is plotted in Fig 8-3. The plot on the left illustrates the linearity of the system for wavelengths ranging from 430 nm to 624 nm. The plot on the right illustrates the linearity of the system for wavelengths ranging from 625 nm to 940 nm. Unsurprisingly, the red and infrared illuminators are more efficient than the ultraviolet and visible light illuminators. Furthermore, the TEMD5080 photodiode is more sensitive to red and near-infrared illumination.

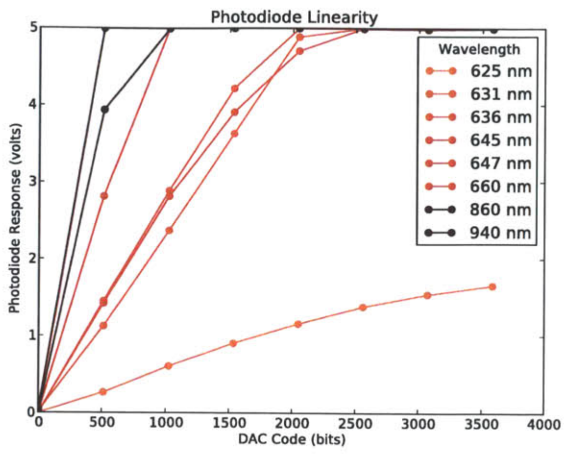
As shown in Fig 8-3a on the left, the shorter wavelength illuminators are clearly bifurcated. Simply, some illuminators are more efficient than others. The more efficient LEDs are clustered towards the top of the plot; the less efficient illuminators, towards the bottom. In the instrument, however, the relative power output of each illuminator is of no consequence and will be corrected via calibration.

8.2.2 Sample Identification

In the first experiment, the transmission spectroscopy instrument pictured in Fig 8-1 is employed to identify samples. In this example, the instrument illuminates the sample with a series of LEDs of varying



(a) Linearity of illuminators and photodiodes on fixture one.



(b) Linearity of illuminators and photodiodes on fixture two.

Figure 8-3: Linearity of the illuminator and photodiode system.

wavelength. The intensity of the transmitted light is measured with a pair of photodiodes. The relation between transmitted intensity and wavelength is stored as a vector. This vector is then used to identify subsequent samples.

Another approach is to adjust the illuminator output until the light falling on the detector reaches some set point. The identification vector is composed of illumination control currents rather than detected photocurrents. In principle, these schemes should be equally effective. In practice, however, sample characterization by illuminator drive currents is almost never used.

8.2.3 Device Drive Currents

To measure samples with the transmission spectrometer pictured in Fig 8-1, LED drive currents must first be determined. In the experiment described in Chapter ??, device currents were programmed to compare each illuminator's spectrum. By contrast, when prescribing LED currents for transmission spectroscopy, the illuminator brightness is set to just below the saturation point of the sensor, assuming samples attenuate incident light. Therefore, the device drive currents were chosen to maximize the light output of each device while keeping their light output below the saturation point of the detector.

Since some light emitting diodes are more efficient than others, some illuminators are run at the maximum available drive current while others are run at more modest currents. Device drive currents were

Manufacturer	Part Number	λ_{center}	DAC Code	Current
Chicago Miniature Light	CMD15-21UBC/TR8	430 nm	4096	45 mA
Kingbright Corp	APT3216QBC/D	470 nm	2048	25 mA
Kingbright Corp	APT3216CGCK	570 nm	4096	45 mA
Lite-On Inc	LTST-C150KGKT	571 nm	4096	45 mA
Lumex Opto Inc	SML-LX1206SYC-TR	590 nm	4096	45 mA
Lite-On Inc	LTST-C150KSKT	591 nm	4096	45 mA
Lumex Opto Inc	SML-LX1206SOC-TR	610 nm	2048	25 mA
Everlight Electronics	QTLP650CRTR	624 nm	2048	25 mA
Kingbright Corp	APTD3216EC	625 nm	4096	45 mA
Lite-On Inc	LTST-C150KRKT	631 nm	1024	12.5 mA
Lumex Opto Inc	SML-LX1206SIC-TR	636 nm	1024	12.5 mA
Panasonic	LNJ218C82RA	645 nm	512	6.25 mA
Stanley Electric Co	HBR1105W-TR	647 nm	1024	12.5 mA
Chicago Miniature Light	CMD11-21SRC/TR8	660 nm	1024	12.5 mA
Osram Opto Semi	SFH 4059-Z	860 nm	512	6.25 mA
Everlight Electronics	IR26-21C/L110/TR8	940 nm	1024	12.5 mA

Table 8.1: Device drive currents employed for transmission spectroscopy measurements.

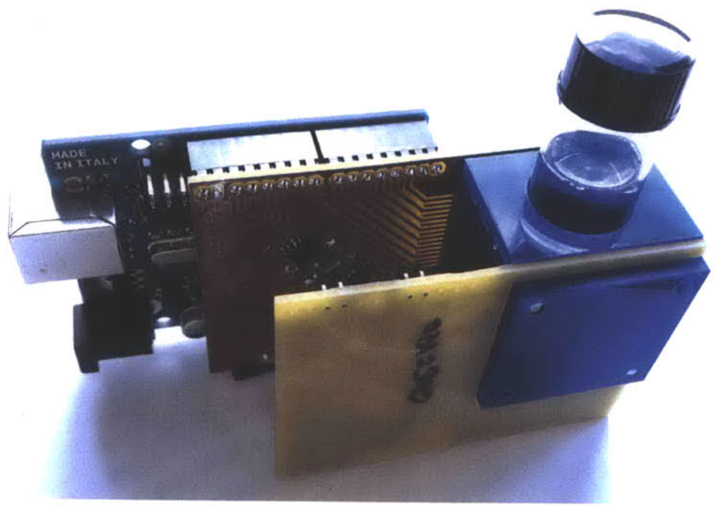
determined with the help of plots Fig 8-3a and Fig 8-3b and through manual adjustment using the graphical user interface discussed in Chapter 4 and Appendix D. The device drive currents for this fixture are enumerated in Table 8.1.

8.2.4 Sample Learning and Discrimination

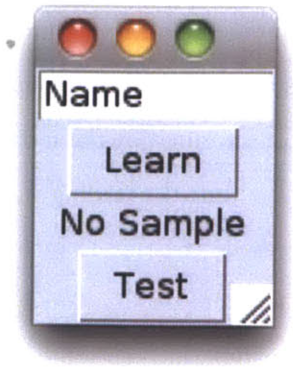
First, a portion of the sample is drawn into a 10 mL borosilicate glass vial. The vial is inserted into the sample holder in the instrument pictured in Fig 8-4a. As previously discussed, the instrument is composed of an illuminator, a plastic sample holder, a detector, and an Arduino. The Arduino, when directed via computer commands, selectively illuminates a combination of the attached light emitting diodes, measuring the incident flux on the photodiodes.

To control the instrument, a graphical user interface, shown in Fig 8-4b was developed. After the sample has been inserted into the instrument, the user enters a sample name in the "Name" field and presses the "Learn" button in order to learn the spectral characteristics of a sample.

The software first takes a "dark" reading with the illumination LEDs off to measure background illumination. Then, the software instructs the instrument to step through each available illuminator. The



(a) Transmission spectroscopy instrument with a sample.



(b) Instrument control GUI.

Figure 8-4: Transmission spectroscopy instrument and the graphical user interface used to control it.

largest reading of the two photodiodes is taken as the measurement of the transmitted light. The software then subtracts the dark reading from the measured transmitted intensities forming a characteristic sample vector.

To identify an unknown sample inserted into the instrument, the user presses the "Test" button. When this button is pressed, the instrument measures the sample similar to when the user presses the "Learn" button. First, the instrument is directed to take a dark reading with the illumination LEDs off. Next, the instrument cycles through each illuminator LED and measures the transmitted light. The vector of the sample to be identified is then compared against the characteristic vectors of known samples.

In this implementation, the sum of squared differences between the candidate and reference samples is computed. The candidate sample is identified by finding the closest reference sample. In this implementation, the use case is sample discrimination. Candidate samples which have not been trained into the system cannot be reliably identified, nor can they be identified as samples which have not been trained. In other implementations, a threshold on the maximum permissible distance or a threshold on identification probability could be used to identify samples which lack training data.

Instrument configuration information, including the location, wavelength, and model number of each illuminator and detector, is stored in a csv file which is loaded when the program is started. With this, the same software can be used with a variety of transmission spectrometers provided their physical configuration is appropriately documented in its corresponding configuration file.

When the graphical user interface is exited, the software stores all of the characteristic vectors in a csv file, similar to the instrument configuration file. The characteristic vectors file stores the name of each sample measured by the instrument. The instrument also stores readings of unknown samples named after the time the reading was taken. The reading is a dark-corrected vector of voltages describing the photodiode response to each of the test wavelengths. The uncorrected measurement and the recorded dark currents are also stored in the file. In addition, this file duplicates some instrument configuration information, including the name and wavelength of each illuminator.

The source code for the transmission spectroscopy graphical user interface and inference algorithms are reproduced in Appendix G.

8.2.5 Sample Learning and Discrimination Examples

Several samples were evaluated in order to measure the performance of the transmission spectrometer. In one test, vials of Coca-Cola Classic, Diet Coca-Cola, vodka, and water were first characterized by the system; the system's discrimination performance was then evaluated.

To discriminate the liquids in this experiment, the samples were illuminated with eight LEDs which had wavelengths ranging from 430 nm to 624 nm. The LEDs were driven with the device currents given in Table 8.1. These were chosen to maximize light output without saturating the photodetectors.

For each sample, a dark-corrected measurement of transmitted optical power for each illumination wavelength is taken. Those measurement vectors describe the spectral characteristics of each sample. When presented with an unknown sample, the system identifies it as the most similar known sample. In these tests, the sum of squared distances is used as the similarity metric.

The vials of samples to be discriminated are shown in Fig 8-5. In the figure, from left to right, the samples contained in the vials are Coca-Cola Classic, Diet Coca-Cola, vodka, water. To the human eye, diet is barely distinguishable from regular Coca-Cola; vodka and water are indistinguishable as well. However, the spectrometer can readily tell the samples apart.

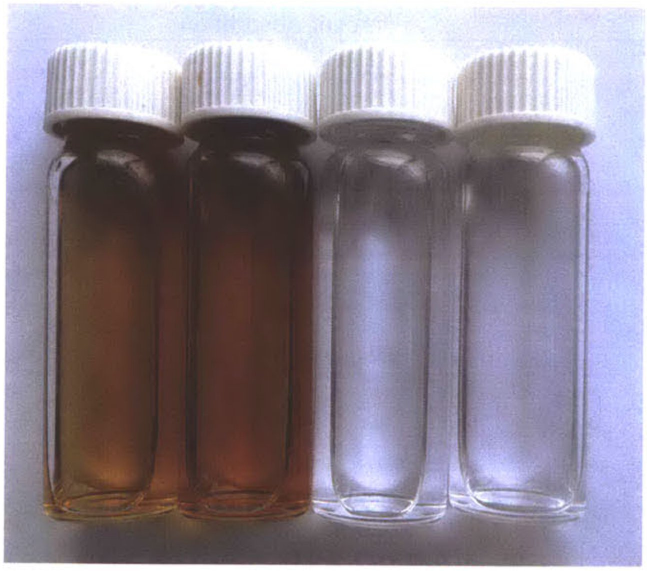


Figure 8-5: Vials of samples. From left to right; Coca-Cola Classic, Diet Coca-Cola, vodka, water. To the human eye, diet is barely distinguishable from regular Coca-Cola, vodka and water are indistinguishable.

Plots of transmitted intensity relative to air versus wavelength are shown in Fig 8-6. In the figure, the upper curves represent the transmission spectra of water and vodka. The lower curves represent the transmission spectra of classic and diet Coca-Cola. In the figure, the markers are colored to represent the wavelength of light the sample is illuminated with.

The plot intensities shown are relative to a null sample, air. The plot at first appears incorrect because some wavelengths have *higher* intensities with a sample than without. Since there is no amplification in the sample (as in a laser cavity), these results appear erroneous. Other factors though, can explain the increase in photocurrent. When there is no sample in the instrument, much of the emitted light is not sensed. This is because there are no collimation optics on either the illuminators nor the detectors. When a sample is placed into the instrument, light can be collimated and focused into the detector, depending on the refractive index of the sample and the shape of the vial. This effect explains why there is sometimes higher photocurrent with a sample than without. This effect could be explicitly exploited in an instrument.

In the plot, it is apparent that diet attenuates all test wavelengths more than regular Coca-Cola. Both vodka and water increase the light falling on the detector because they form a rudimentary collimating lens. Water and vodka have wavelength dependent indexes of refraction, so for some wavelengths, vodka

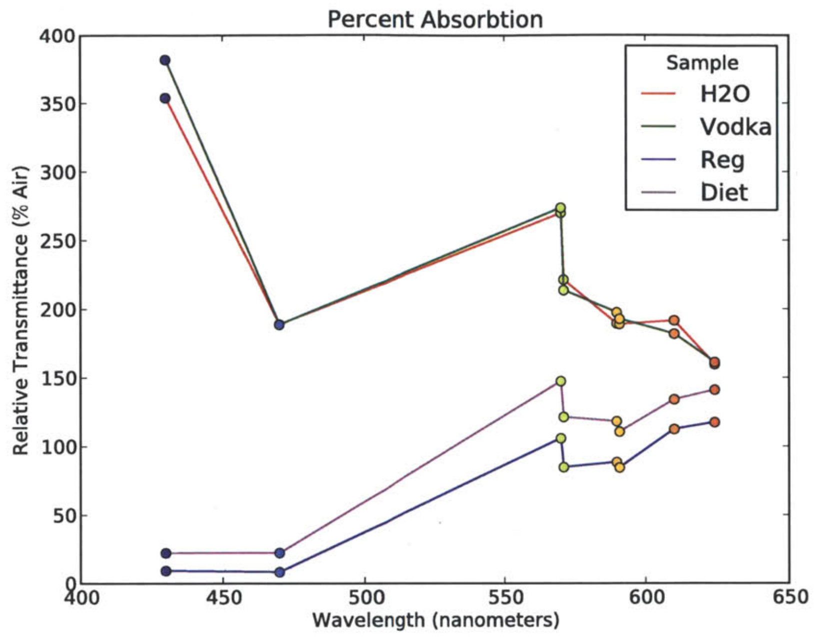


Figure 8-6: Percent transmission relative to air. The two upper curves denote water and vodka, the lower curves represent Diet and Coca-Cola Classic. The markers are colored to represent the illumination wavelength.

transmits more light. For others, water transmits more light. This complex relationship increases the robustness of the measurements because they become independent to slowly varying sources of error. In the case of cola discrimination, an increase in ambient light between readings could corrupt the measurement. Because of the nature of their spectral response, vodka and water discrimination would be resistant to such contamination.

This instrument was informally tested in a variety of environments with a mixture of samples. In all tests, the instrument never struggled to discriminate between Diet Coca-Cola, Coca-Cola Classic, water or vodka. The instrument operated well even under varying light conditions. For example, the instrument could be trained with a sample indoors and then used to discriminate samples outside. The only cases where the instrument struggled were those where the photodetectors were completely overwhelmed by ambient illumination. In those cases, dark correction could not be used to overcome ambient light, and the readings for samples were inconsequential. Light baffles, carefully engineered mechanical fixtures, and appropriately sized sample vials largely eliminate effects from ambient illumination.

8.3 Supervised Transmission Spectroscopy

In the case of supervised transmission spectroscopy, the samples to be discriminated have well characterized spectra. Here, rather than developing a general purpose instrument, a special purpose device is developed. This device measures only spectral features that are pertinent to its prescribed use case. To expand upon the use cases presented in the previous section, a pair of instruments are developed. One such instrument is purpose built to differentiate between Classic and Diet Coca-Cola, and the other is designed to discriminate vodka from water.

8.3.1 Sample Wavelength Selection

When selecting spectral features for sample discrimination, a number of factors are considered. These design factors are discussed in greater detail in Chapter 2. The most important characteristic of a spectral feature is its effectiveness. For a designer, it is most important to select features which maximally separate the spectral vectors to be discriminated. Unfortunately mitigating factors complicate this process.

If the particular use case necessitates it be particularly compact, inexpensive, low-cost or high accuracy, the wavelengths which can be searched over are reduced. If the device must be compact but also very accurate, the designer might choose a holographic or integrated optics spectrometer. If the design must be low cost, the best solution might be to use solid state illumination. Each of these considerations influences the available features which can be used to discriminate between samples.

Several of the designs in this chapter are constructed by first considering the ancillary constraints on the design and then choosing the most effective spectral features to discriminate the samples. In each of the examples in this chapter, the set of features are composed of the spectra of light emitting diodes. Fig 8-6 can be used to choose which LEDs best discriminate vodka from water and diet from classic Coca-Cola.

In these instruments, LEDs with wavelengths 430 nm, 571 nm, and 610 nm are used to discriminate vodka from water. LEDs with wavelengths 570 nm, 590 nm, and 624 nm are used to discriminate diet from classic Coca-Cola.

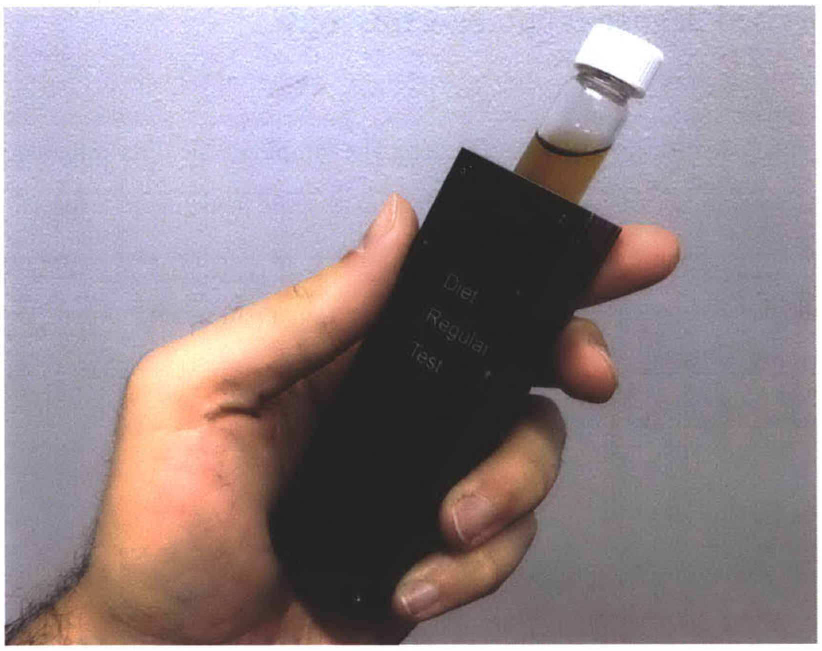


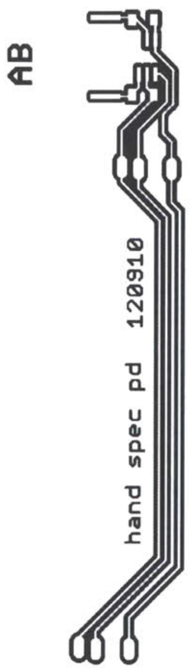
Figure 8-7: Photo of completed hand-held spectrometer. This model discriminates between Diet and Classic Coca-Cola.

8.3.2 Hand-Held Spectrometer

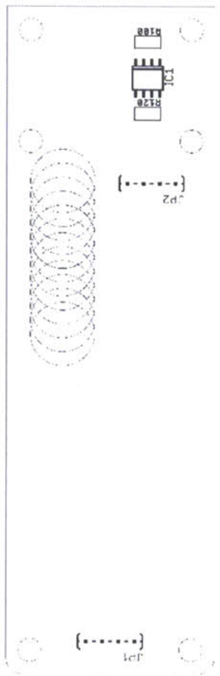
A common hardware platform was developed for both the vodka from water discriminator and Diet from Classic Coca-Cola discriminator. This spectrometer is designed to be hand-held, battery operated and accept 10 mL vials. In the design, the sample can be probed with up to six LEDs in 1206 packages. The transmitted light is measured with two TEMD5080X01 photodiodes. The architecture is similar to, but more integrated than the spectrometer design pictured in Fig 8-4a.

The electrical system is composed of two main circuit boards: the illuminator board and the detector board. The detector board contains a pair of TEMD5080X01 photodiodes and an OPA2350 dual operational amplifier. Each photodiode is connected to an op-amp configured as a transimpedance amplifier. The transimpedance amplifier is designed to have a gain of 10 µA/V. The output of the transimpedance amplifier is pinned out to a connector. This connector is designed to accept a wiring harness from the illuminator circuit board.

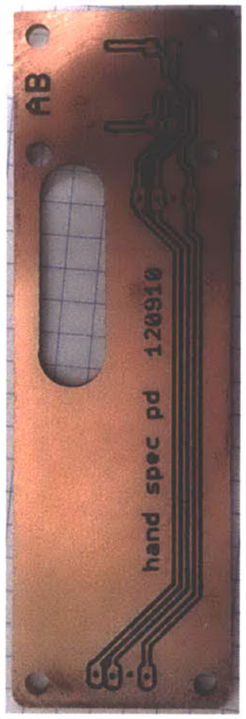
The illuminator board carries the set of illuminators which probe the sample, the user interface indicators and pushbutton, power supplies, the micro-controller and its support circuitry. Unlike the illuminator



(a) Hand-held spectrometer illuminator circuit board top layer artwork.



(b) Hand-held spectrometer illuminator circuit board placement diagram.

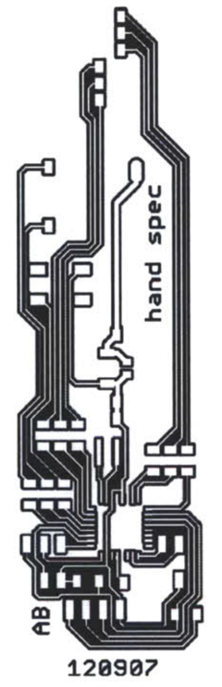


(c) Photo of fabricated hand-held spectrometer circuit board.

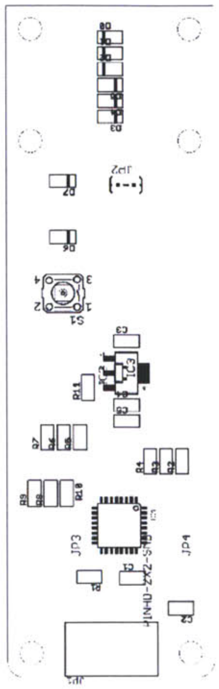
Figure 8-8: Hand-held spectrometer top layer artwork, placement diagram and photo of fabricated and partially populated circuit board.

board in the unsupervised spectrometer, the current through the LEDs is not run-time programmable, but rather set via series resistors soldered to the board. The micro-controller, an Atmel ATMega168, and its support circuitry, are designed similarly to the Anduino discussed in Appendix C making them Arduino-compatible. The processor runs at 16 MHz derived via a ceramic resonator. The circuit is designed to operate from a single 9 V battery regulated to 5 V for the ATMega. Similar to the Anduino described in Appendix C, this circuit board lacks a USB-to-serial converter, so a standard FTDI header is included for computer communication and programming. The case has an optional cutout to allow access to the FTDI header for in the field programming.

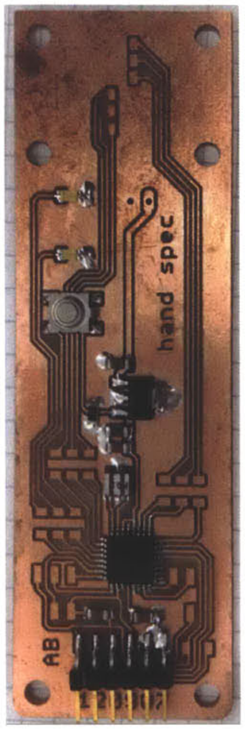
The hand-held spectrometer can be operated in a number of ways. In one mode, the spectral characteristics of each sample are measured using some other instrument. Then, using knowledge of the spectrum of the illuminator LEDs, a spectral response vector is computed and uploaded from a computer to the



(a) Hand-held spectrometer illuminator circuit board top layer artwork.



(b) Hand-held spectrometer illuminator circuit board placement diagram.



(c) Photo of fabricated and partially populated hand-held spectrometer circuit board.

Figure 8-9: Hand-held spectrometer top layer artwork, placement diagram and photo of fabricated and partially populated circuit board.

instrument. Alternatively, the instrument can serve as the spectrometer. When a sample is placed in the instrument, if the "Test" button is held down for three seconds, the instrument enters a learn mode. The button can be pressed again to toggle between the two samples. The button is held for another three seconds to save the response vector.

In limited informal testing, both the vodka/water and diet/regular spectrometers easily discriminated among samples. The units are conveniently sized and can be easily fitted into a purse or coat pocket. There is, of course, room for improvement over these first prototypes. For one, there are no provisions for a battery holder, and the 9 V battery is simply double-sided taped onto the inside of the case. Furthermore, to replace the battery, the user has to completely disassemble the device using a screwdriver. Disassembly is not difficult; however, reassembly can be difficult due to the alignment of the indicator light pipes and pushbutton extension. Aggravating the issue, there is no physical mechanism to completely power off the device. That said, each battery is estimated to provide over 500,000 measurements on a single charge and the batteries in the test units have not yet been replaced, even after thousands of uses. Finally, the board-to-board harness is inconveniently placed, and should be re-engineered in future hardware revisions.

Additional materials, including complete schematics, board layouts, population diagrams, and source code for these spectrometers are reproduced in Appendix G.

8.4 Conclusions

Similar to the unsupervised spectrometer, the hand-held supervised spectrometer is resistant to external noise sources, such as ambient illumination. During informal testing, the spectrometers performed as designed, easily discriminating the desired samples. Though the devices are tiny by comparison to a laboratory spectrometer, using modern manufacturing methods the spectrometers could easily shrink in size by an order of magnitude. Similarly, though the cost of a single prototype is less than $20, in thousand unit quantities, these devices could easily be manufactured for a few dollars. That said, for some applications, these prototypes could turn into plausible products and demonstrate the viability of the concept of reduced dimensionality simple spectral sensors.

Chapter 9

Discussions & Conclusions

Spectroscopy is a powerful tool in material identification, characterization and discrimination. Unfortunately industrial and laboratory spectrometers are typically very large, costly, and inconvenient. The aim of this thesis is to broaden the awareness and appeal of spectroscopic sensing modalities by exploring specialized, rather than general purpose instruments. Rather than sensing the entire spectrum, these devices work by observing just the particular spectral features needed to perform identification or discrimination. This approach greatly simplifies the instrument reducing the cost, size, power consumption, and analysis complexity by many orders of magnitude.

9.1 Further Considerations

A number of considerations must be made during the design of these sensors. The most central issue is discrimination wavelength. If, for example, samples A and B are to be differentiated, the designer must choose some set of wavelengths in the spectrum of each sample which allows them to be discriminated. Spectral features should be chosen that are not time varying, are resistant to external factors (such as sunlight), and are consistent with the entire sample, as some samples may be heterogeneous. In some cases, such spectral features may not exist - their spectra may appear, to some spectral basis, identical. In other cases, the spectra of the samples may vary, but the instrument may not be sensitive enough to discriminate them.

Since the principle application of this work is consumer goods, cost plays a large factor. That said, some consumer applications may warrant more expensive instruments. In the case of counterfeit detection, a costlier instrument may be justified, such as in the case of knock-offs of luxury clothing and accessories. A manufacturer could offer counterfeit detection instruments at no cost to the patron, as an incentive to purchase their product. This work could also see application in industries such as defense and medicine, which are more cost tolerant.

Similarly, the size of the instrument must be considered. In some cases, the user may wish to transport the device with them, say, in their pocket, purse or backpack. In other cases, the instrument could be incorporated into a piece of furniture, such as a bar, kiosk or podium. Other applications may require the instrument to be integrated into a cup, a scale, or a cell phone.

If the instrument is mobile, the power consumption of the device becomes important. If the sensor is made part of, for example, an extra-solar space probe, cost and size requirements may be relaxed, but minimizing power consumption would be essential. Likewise, mobile surveying equipment can be rather costly, but would have to operate for days or weeks on battery power.

Depending on the application, the instrument may be subject to a number of environmental factors. Airborne devices are subject to temperature extremes and very high forces and vibrations. Field instruments, such as monitoring buoys, must be waterproofed. Space-borne spectrometers must be able to withstand cosmic radiation. Instruments intended for long term monitoring applications, such as those at a wastewater treatment plant, require some care to ensure their sensing windows remain clean of debris. In other cases, the sample might be hot, toxic or corrosive necessitating exotic window materials such as fused sapphire or alumina glass.

The required detection rate can influence instrument design in unexpected ways. For example, if the user wishes to avoids land mines, it is better if the instrument errs on the side of false positives rather than false negatives. If there is the slightest chance of a land mine, the instrument reports that the area is unsafe. If, instead, the operator wants to find land mines to clear an area, it may be better to err on the side of false negatives. For a seemingly identical detection task, two different systems are be designed.

(a) A doctor tests medicine. (b) A patron's drink is safe. (c) A patient's blood is tested.

Figure 9-1: Potential use cases.

9.2 Use Cases

This work envisions a number of potential use cases ranging from medical, food science, defense, entertainment, and agriculture. One such case is validation of vaccines, as in Fig 9-1a. Presently, a shipment of medication may have sensors to detect violent handling, exposure extreme temperatures, light or oxygen. Those sensors detect the presence of effects which may cause spoilage rather than testing the vaccine itself to determine if it was spoiled. It's possible to envision a scenario whereby the vaccine vial is inserted into an applicator which tests the vaccination to ensure its quality before allowing injection. A similar detector could be used to check if the vaccine is genuine before administration.

Furthermore, in food science, there are many food and drink items in which tampering is rampant. A sensor could be conceived to detect a particular food additive, such as peanuts or dairy, to warn an allergic patron. It may also be possible to construct a sensor to detect if food or drink has been dosed with an illicit substance, such as Rohypnol. Similarly, sushi, wine, and alcohols are all sometimes subject to impersonation by unscrupulous vendors. One could imagine a case where a luxury alcohol is promoted by encouraging the customer to verify its authenticity table-side.

9.3 Conclusions

In this work, the anatomy of such specialized sensors is explored. A thorough discussion of illuminators, current sources, photodetectors, photodiode amplifiers, control systems and part selection is included.

In these chapters, instruments are designed and fabricated. Their tradeoffs are enumerated and discussed. Finally, these building-blocks are combined to construct several working prototypes which are informally characterized.

The author is hopeful this work will broaden the appeal of such spectroscopic measurements and techniques by demonstrating the operation through several practical designs. In addition, because these sensors have specialized rather than general purposes, they are significantly less complex, less expensive, and far simpler in operation than their commercial counterparts, further increasing their appeal.

Appendix A

Building an Illumination Toolbox: Scraping Digikey

A.1 Introduction

Most of the applications presented in this thesis focus on solid state LED illumination. Despite the typical constraints of size and power consumption, the illuminator's emitted wavelength firmly trumps the other factors. When building a toolbox of LED illuminators, it is important to sort by emitted wavelength, rather than package, lens type, transmission angle or other confounding parameters.

Several tools were constructed to catalog and search all available LED illuminators from Digikey [10], a popular electronics retailer. These tools, `scrape.py` and `plot.py` are detailed below.

A.2 `scrape.py`

`scrape.py`, systematically builds a SQL database of every part, when presented with a valid Digikey url (several examples of valid urls are given in the source). For this application, `scrape.py` was used to index every LED illuminator that Digikey sold (available or not), but it is flexible enough to index any part offered by Digikey.

In particular, the categories of LEDs considered for this work include (all in the optoelectronics category):

- LEDs - <75mA, Discrete: General purpose indicators.
- LEDs - >75mA, High Brightness, Power: High brightness illuminators.
- Infrared Emitters: Infrared illuminators and signal devices.

As of July 10, 2012, these three categories contained 23,852 unique products, 7,872 were listed as in stock. On digikey.com, these products are listed over nearly five hundred different web pages. Though for most categories, Digikey boasts impressive search tools, often some parameters are missing and comparing across categories is near impossible. `scrape.py` was authored to facilitate deep parameter comparison and indexing of all products across varied categories.

`scrape.py`, reproduced below, downloads the entire Digikey directory given some root url. `scrape.py` uses the powerful package `BeautifulSoup` [36] to traverse the markup on the webpage. The product table is then stored in a table in a SQLite database. The table is named after the child url and the date the table was created.

Running `scrape.py`, on "LEDs - <75mA, Discrete" takes between 30 and 45 minutes on a late 2008 MacBook Pro, dependent on connection speed.

A.2.1 `scrape.py` Source Code

```
#!/usr/bin/python
##############################################################################
# scrape.py
#
# Andy Bardagjy
# 5/10/2012
#
# This code may be vulnerable to SQL injection attacks, but "safer"
# code tended to be buggy.
#
##############################################################################

import sys
import urllib2
from bs4 import BeautifulSoup
import lxml
import sqlite3 as sqlite
import datetime
import string
import unicodedata

#import pdb; pdb.set_trace() # This is for setting the debug environment

#database name, TODO pass it on the command line
db = sqlite.connect('dk.sqlite')
c = db.cursor()

# Default urls for scraping 75ma discrete LEDs
root_url = 'http://search.digikey.com'
child_url = '/us/en/cat/optoelectronics/leds-75ma-discrete/524729/'

# TODO maybe in the future I'll strip punctuation and whitespaces
# to deal with funky spacing e.g
# url.strip(string.punctuation).strip().split('/')
try:
    url = sys.argv[1]
    if url[0:4] == 'http':
        if url[0:len(root_url)] == root_url:
            child_url = url[len(root_url):]
        else:
            print('Strange url, did DK change its scheme?')
except IndexError:
    pass

print("Parent URL: %s" %root_url)
print("Child  URL: %s" %child_url)

page = urllib2.urlopen(root_url + child_url)
soup = BeautifulSoup(page, 'lxml')

# Make a new table with name $child_url_$DATE
today = datetime.datetime.today()
datestring = today.strftime('%Y%m%d')
tmpurl = child_url.strip('/').strip(' ').split('/')[-2]
tmpurl = tmpurl.replace('-','_')
tablename = tmpurl + '_' + datestring
sql = 'CREATE TABLE %s(' %(tablename)

```

```python
header = []
ptable = soup.find('table', id='productTable')
for th in ptable.thead.tr.find_all('th'):
    # If there's only one thing in that table element
    if len(list(th.children)) == 1:
        text = th.get_text()
    if text == "": text = th.img['alt'] #This is to capture datasheets
    if text == "": text = "No Entry"
    # Convert from unicode to "normal" string
    try:
        text = unicodedata.normalize('NFKD',text).encode('ascii','ignore')
    except TypeError:
        pass
    # Get rid of whitespaces and punctuation
    text = text.translate(string.maketrans('',''),string.punctuation+' ').strip()
    header.append(text)
    sql += text + ' TEXT,'

sql = sql[:len(sql)-1] + ')' # To remove trailing comma
c.execute('DROP TABLE IF EXISTS %s' %(tablename))
#print(sql)
c.execute(sql)
sql=''

# Loop through all pages
page_count = 0
while True:
    page = urllib2.urlopen(root_url + child_url)
    soup = BeautifulSoup(page,'lxml')
    ptable = soup.find('table', id='productTable')

    # Loop through all rows in the product table on that page
    row_count = 0
    for current_row in ptable.tbody.find_all('tr'):
        # Craft a sql statement to put that row into the table
        sql = 'INSERT INTO %s VALUES(' %(tablename)
        row = []
        for row_element in current_row.find_all('td'):
            # If there's nothing in the table
            if len(list(row_element.children)) == 0:
                text = "No Entry"
            # If there's only one thing in that table element
            elif len(list(row_element.children)) == 1:
                text = row_element.get_text().strip()
                # If there was no text
                if text == "":
                    # It might be an image? Get src url
                    if row_element.find('img')['src'] > 0:
                        text = row_element.find('img')['src']
                        # Is it a datasheet icon? Get url of datasheet
                        if row_element.find('img')['src'].find('datasheet') > 0:
                            text = row_element.find('a')['href']
                        # Is it a relative URL?
                        if text[0:4] != 'http':
                            text = root_url + text
                    # It might be a link? Get href
                    elif row_element.find('a')['href'] > 0:
                        text = row_element.find('a')['href']
                        # Is it a relative URL?
                        if text[0:4] != 'http':
                            text = root_url + text
                    else:
                        "Unknown (not img or href) element"
            # More than one thing in the table element
            else:
                # Well usually the text of the first thing is what you want
                try:
                    text = row_element.contents[0].get_text().strip()
                except AttributeError:
                    text = row_element.contents[0].strip()
                if text == "":
                    text = "Unhandled Case"
            # TODO Should get rid of special characters and whitespace here
            row.append(text.strip())
            sql += '?,'
            text = "Unhandled Case"

        print("Page " + str(page_count) + " Row " + str(row_count))
        #print(row)
        sql = sql[:len(sql)-1] + ')' # To remove trailing comma
        #print(sql)
        c.execute(sql,tuple(row))
        sql = ''
        row_count += 1

    if soup.find(text="Next") > 0:
        child_url = soup.find(text='Next').parent['href']
        page_count += 1
    else:
        break
```

```

# Must do this before exiting or the changes will not be written.
db.commit()
```

A.2.2 `scrape.py` Errata and Future Work

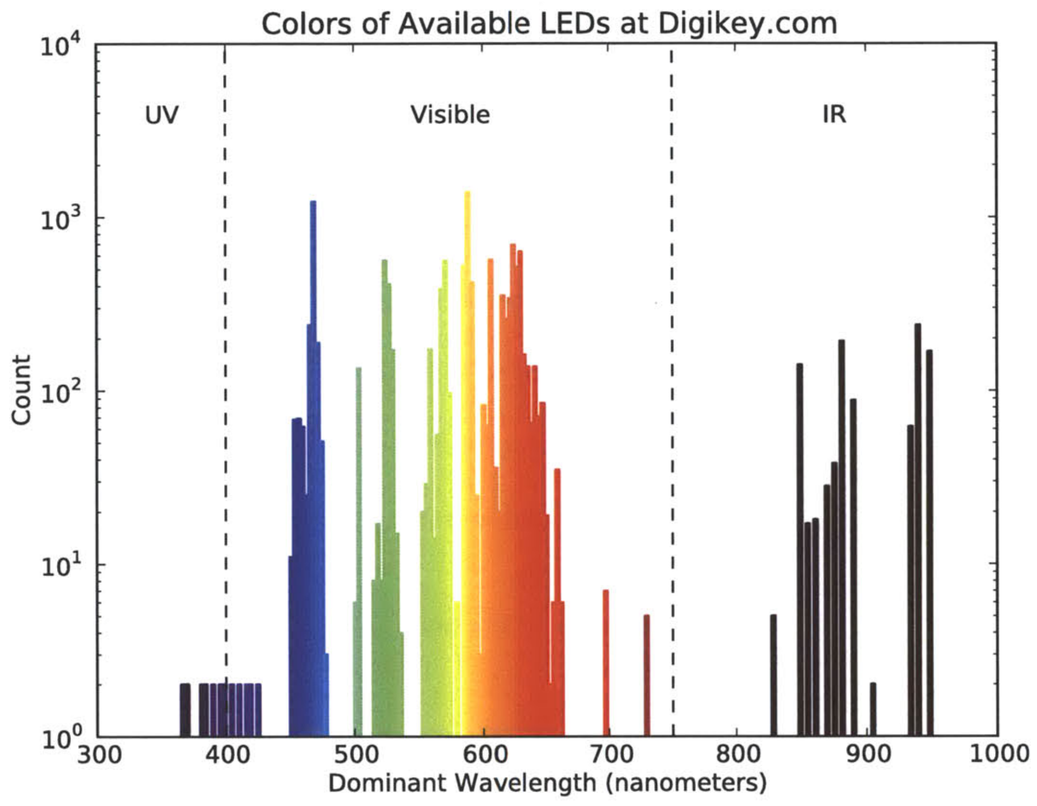


Figure A-1: Colors of available LEDs at Digikey.com.

Though `scrape.py` was used successfully for many months during the spring and summer of 2012, some improvements and modifications remain outstanding. In particular, SQL commands are not "sanitized" and thus `scrape.py` is vulnerable to SQL injection attacks. While this may sound fanatical, some websites include SQL injections as a matter of course in their HTML source.

The script could presumably be tremendously sped up by launching multiple threads which query different webpages. A nice addition would be increased command line arguments, in particular, the ability to specify database name, and perhaps an array of URLs to be fetched.

In the future, with this script, time varying prices and availability of components could be explored. This could provide some small insight into the mechanics of the electronics industry such as the impact of natural disasters and big movers on price and availability of components. With careful study, one could use this omniscient knowledge about this corner of the electronics industry to speculate on electrical components similar to the stock market.

A.3 Visualizing Every LED Available at Digikey.com: `plot.py`

As a simple example, after scraping all LEDs listed on Digikey, their nominal wavelengths were plotted with the script `plot.py`. `plot.py`, reads the scraped data from the database, then using matplotlib and pyplot, graphs the count of available LEDs versus their wavelength (inverse frequency).

A.3.1 `plot.py` Source Code

```
#!/usr/bin/python
##############################################################################
# plot.py
#
# Andy Bardagjy
# 5/14/2012
#
##############################################################################

import sqlite3 as sqlite
import sys
import matplotlib.pyplot as plt
from wavelen2rgb import wavelen2rgb
#import pdb; pdb.set_trace()

dbname = 'dk.sqlite'
db = sqlite.connect(dbname)

c = db.cursor()

tables = []
```

```

print('The tables in the database %s are' %(dbname))
for table in c.execute('SELECT name FROM sqlite_master \
                        WHERE type=\'table\' ORDER BY name'):
    tables.append(table[0])

wavelengths = []
for table in tables:
    print(table)

    # This block of code determines if we should look at the
    # Wavelength or WavelengthDominant column
    c.execute('SELECT * FROM %s' % (table))
    cols = [tuple[0] for tuple in c.description]
    col = 'Wavelength'
    try:
        cols.index(col)
    except ValueError:
        col = 'WavelengthDominant'

    for element in c.execute('SELECT %s FROM %s' % \
                                (col, table)):
        element = element[0]
        for entry in element.split():
            if entry[len(entry)-2:len(entry)] == 'nm':
                wavelengths.append(float(entry[:len(entry)-2]))

fig = plt.figure()
ax = fig.add_subplot(111)
n, bins, patches = ax.hist(wavelengths, bins=200, log=True)

for patch in patches:
    wavelen = patch.xy[0]
    if wavelen > 780:
        wavelen = 780
    elif wavelen < 380:
        wavelen = 380
    # So wavelen2rgb returns whole digits, e.g. if MaxIntensity is set to 1,
    # it rounds to either 0 or 1. So, scale to get "floats"
    color = wavelen2rgb(wavelen,MaxIntensity=255)
    color = map(lambda x: x/255.0, color)
    patch.set_facecolor(color)
    patch.set_edgecolor(color)

ax.set_title("Colors of Available LEDs at Digikey.com")
ax.set_xlabel("Dominant Wavelength (nanometers)")
ax.set_ylabel("Count")

plt.ylim([1,10000])
ax.set_autoscaley_on(False)
ax.vlines(400,1,100000, color='black', linestyles='dashed')
ax.vlines(750,1,100000, color='black', linestyles='dashed')

ax.text(575, 3500, "Visible", horizontalalignment='center', fontsize=12)
ax.text(350, 3500, "UV", horizontalalignment='center', fontsize=12)
ax.text(875, 3500, "IR", horizontalalignment='center', fontsize=12)

#plt.show()
fig.savefig('count.pdf')
```

A.4 Epilogue and Notes

Many resources were helpful during development. The Firefox extension, sqlite-viewer was and invaluable debugging tool during development. Finally, the BeautifulSoup screen-scraping library was tremendously useful for parsing downloaded html.

In the future, this script may prove to be a powerful lens into the mysterious world of electronic component pricing and availability. By tracking changes in component price, it may be possible to discover latent electronics development, possibly allowing market prediction. In addition, prediction of component price could prove lucrative to potential buyers.

Appendix B

Automated Measurements

B.1 Introduction

During development of several circuits and sensors over the process of this thesis, frequently, there arose the necessity to automate measurements. This section outlines the generic methodology used for automated measurements, including software and hardware used for those measurements.

B.2 Methodology

As a proof of concept to verify proper computer controlled operation, the linearity of an Agilent E3631A triple output DC power supply was measured with an Agilent 34401A 6 ½ digit multimeter [44, 45]. Both the E3631A and the 34401A can be computer controlled via either GPIB or RS-232 and communicate via VISA, the Virtual Instrument Software Architecture, a robust command set layered on top of some physical layer (in this case GPIB or RS-232) [38].VISA, while largely an open spec, is often used via National Instrument's proprietary - but free - implementation [30].

National Instruments VISA, hereafter referred to as NI VISA, is a free, but proprietary multi-platform (Windows, Linux and OSX) implementation of the VISA spec. NI VISA includes libraries and headers which can be linked into (typically C++) applications. In addition, NI VISA includes various management utilities including NI VISA Configuration to add additional devices, NI VISA Server to use VISA devices remotely, and a driver wizard.

B.2.1 PyVISA: Python NI VISA wrapper

PyVISA wraps the NI VISA libraries using ctypes to allow Python control of VISA devices. Unfortunately, at the time of this writing, PyVISA is not officially supported under Mac OS X. Thankfully, with some minor modifications, PyVISA can be run under OS X (10.6.8 in this case).

Before starting modifications, the libraries and toolkits required were installed. First, NI VISA version 5.1.2 for Mac OS X was downloaded and installed from ni.com. Next, in the user's home directory create a file .pyvisarc with the contents

```
[Paths]
VISA library: /Library/Frameworks/VISA.framework/VISA
```

Then, once PyVISA is downloaded and uncompressed, easy_install is used to install the toolkit.

Unfortunately, it is discovered that NI VISA 5.1.2 for OS X is compiled for 32 bit systems by `$ file /Library/Frameworks/VISA.framework/VISA` which returns i386 not x86_64. By default, Python on modern OS X defaults to 64 bit mode, which is incompatible with NI VISA 5.1.2. If in Python, `import visa` will throw several errors such as `no suitable image found` and `no matching architecture in universal wrapper` which are indicative of an architecture mismatch in ctypes.

The solution is to run Python in 32 bit mode when using PyVISA. In OS X, the environment variable VERSIONER_PYTHON_PREFER_32_BIT can be asserted by `$ export VERSIONER_PYTHON_PREFER_32_BIT es.` To force Python to start in 32 bit mode, this line could be placed in the users .bashrc or .profile. Unfortunately still, many useful Python packages (such as PyLab) are only distributed as 64 bit binaries. It would be best if the user could select 32 or 64 bit Python operation.

One way to do this is to create an alias for 32 bit Python by adding the line `alias python386='arch -i386 python'` to the users .bashrc or .profile.

To force a script to execute under 32 bit Python, rather than starting the script with the line #!/usr/bin/python, the script is headed by #!/usr/bin/env arch -i386 python. Unfortunately many packages are do not ship with 32 bit versions (e.g., numpy and pylab) which forces separate threads or separate scripts, to handle VISA instruments. This may become tiresome if interactive communication with instruments is desired.

B.2.2 Hello World with pyvisa

As a "Hello World" with pyvisa, many instrument programming manuals suggest querying the *IDN? identity of the instrument. Below is a short code snippet querying the identity of the Agilent 34401A 6 ½ digit multimeter.

In this example, the Agilent 34401A is connected to the computer via a Trendnet TU-S9 RS-232 USB serial port adapter based around the PL-2303 chipset. On many operating systems this adapter is natively supported, on others drivers are available from the Prolific website.

The NI-VISA Configuration application lists adapters which are available to VISA. Serial devices appear as ASRLx and GPIB adapters appear as GPIBx. The *IDN? VISA command is used to determine the identity of the connected instrument. In the following example, the Agilent 34401A is connected to the ASRL1 adapter.

```
#!/usr/bin/env arch -i386 python
# PyVISA calls NI VISA which is compiled as 32 bit, so run python in 32 bit

from visa import *

inst = SerialInstrument('ASRL1',
                        baud_rate = 9600,
                        data_bits = 7,
                        stop_bits = 2,
                        parity = even_parity,
                        term_chars = CR+LF,
                        end_input = term_chars_end_input,
                        send_end = True,
                        delay = 2
                        )
```

```

inst.write('SYST:REM')
inst.write('*RST')
sleep(10)
inst.write('SYST:REM')
print inst.ask('*IDN?')
```

If everything is operating correctly, the script should print `HEWLETT-PACKARD,34401A,0,11-5-2`, depending of course, on the installed firmware version.

B.2.3 Agilent E3631A Linearity

As a longer proof of concept of automated measurements, the Agilent 34401A 6 ½ digit multimeter is used to measure the linearity of the Agilent E3631A triple output DC power supply. In this example, the six volt output of the E3631A is set to some voltage, and the output is measured with the 34401A. The measured and set voltages are stored in a CSV file for later processing.

Measurement Source Code

```
#!/usr/bin/env arch -i386 python
# lin.py
# Andy Bardagjy
# July 12,2012
#
# PyVISA calls NI visa which is compiled with 32 bits,
# so run python in 32 bit with #!/usr/bin/env arch -i386 python
#
# Both Agilent 34401A and Agilent E3631A are controlled via serial
# Use NI-VISA Configuration to find which ASRL they are connected to
#

from visa import *
import time
import csv

# CSV open file
f = open('out.csv', 'w')
writer = csv.writer(f)

```

```
# Agilent 34401A
dmm_addr = 'ASRL1'
dmm = SerialInstrument(dmm_addr,
                        baud_rate = 9600,
                        data_bits = 7,
                        stop_bits = 2,
                        parity = even_parity,
                        term_chars = CR+LF,
                        end_input = term_chars_end_input,
                        send_end = True,
                        delay = 1
                        )

# Set it up for remote access, reset to factory, remote again
print "Resetting DMM"
dmm.write('SYST:REM')
dmm.write('*RST')
time.sleep(20)
dmm.write('SYST:REM')

print dmm_addr + ": " + dmm.ask('*IDN?')

# Agilent E3631A
pwr_addr = 'ASRL4'
pwr = SerialInstrument(pwr_addr,
                        baud_rate = 9600,
                        data_bits = 7,
                        stop_bits = 2,
                        parity = even_parity,
                        term_chars = CR+LF,
                        end_input = term_chars_end_input,
                        send_end = True,
                        delay = 1
                        )

# Set it up for remote access, reset to factory, remote again
print "Resetting Power Supply"
pwr.write('SYST:REM')
pwr.write('*RST')
time.sleep(20)
pwr.write('SYST:REM')

print pwr_addr + ": " + pwr.ask('*IDN?')

# Loop from 0 to 6 volts in 10mV increments cyc times
```

```
cyc = 15

voltages = []
volt = 0
for cnt in range(600):
    voltages.append(volt)
    volt = volt + 0.01

writer.writerow(voltages)

for I in range(cyc):
    tarr = []
    for J in range(len(voltages)):
        setpoint = voltages[J]
        # Set the power supply to the setpoint
        pwr.write('APPL P6V, ' + str(setpoint) + '; OUTP ON')
        # Let the output settle 1 seconds
        time.sleep(1)
        # Measure the output
        meas = dmm.ask('MEAS:VOLT:DC? 10, MAX')
        tarr.append(meas)
        # Tell the user what's up
        print("Cycle: " + str(I) +
              " set:" + str(setpoint) +
              "V meas:" + str(meas) + "V")
        # Turn off the output
        pwr.write('OUTP OFF')
    # Write tarr to the file
    writer.writerow(tarr)

# Close the file
f.close()
```

Next, the set voltage is compared to the output voltage and plotted alongside the output voltage tolerance specification.

```
#!/usr/bin/python
# plot.py
# Andy Bardagjy
# 7/13/2012
#
# Grabs stored data csv from lin.py and plots it
#

```

```
import csv
from pylab import *

f = open('out.csv', 'r')
reader = csv.reader(f)
cols = zip(*reader) # transpose reader so it's columnwise

# Measured data
data = []
# Set voltage
setv = []
# Tolarance array (according to the datasheet)
tol = []
# Every 100mV
for col in cols[::10]:
    # Build the set voltage array
    setv.append(float(col[0]))
    # Build the tolarance voltage array
    tol.append(float(col[0])*0.001+0.005)
    # Build the linearity array
    data.append(map(lambda x: float(x) - float(col[0]), col[1:]))

# Close the CSV file
f.close()

# Figure and axis
fig = figure()
ax = fig.add_subplot(111)
ax.boxplot(data)

# Titles and labels
ax.set_title("Agilent E3631A P6V Linearity")
ax.set_xlabel("Set Voltage (volts)")
ax.set_ylabel("Deviation from Set Voltage (volts)")

# Show every 500mV on the axis
axlab = map(lambda x: x if x % 0.5 == 0 else "" ,setv)
ax.set_xticklabels(axlab, rotation = 45)
ax.fill_between(range(1,len(tol)+1),
                tol,
                tol*(-1*ones(len(tol))),
                facecolor='yellow',
                alpha='0.25')

# Save the figure
```

```
fig.savefig('lin.pdf')

# Display the figure
show()
```

Linearity Results

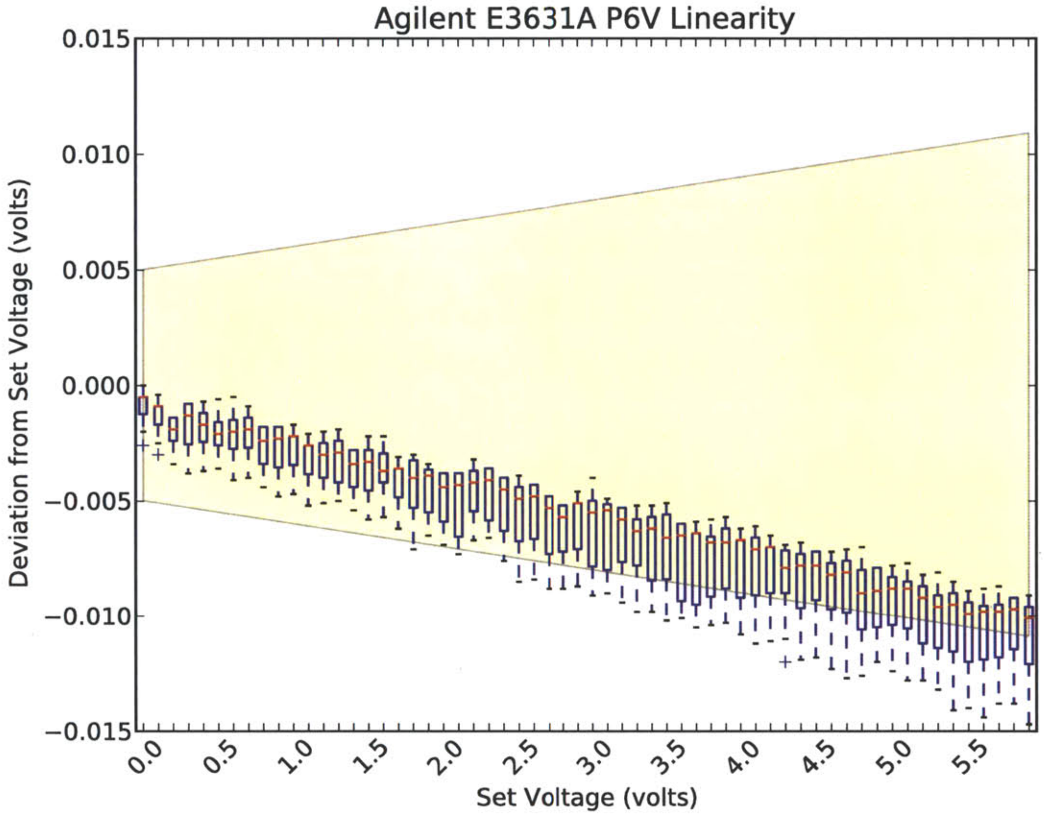


Figure B-1: Linearity of the Agilent E3631A.

Despite both the Agilent 34401A and E3631A being both over ten years old, assuming the 34401A is within its calibration specifications, the E3631A is still operating within factory tolerances (indicated by the yellow trapezoid).

The error bars indicate the upper and lower quartiles, and the red bar indicates the average reading. Outliers are denoted by "+". The power supply was cycled overnight, 15 times per set voltage. The average reading falls within the manufacturer specifications. There appears to be some slight latent structure in the deviation from the set voltage which may indicate range switching or some other mechanism within the power supply. Still, with a maximum deviation from the set point no greater than 10mV, the instrument is quite accurate.

B.2.4 GPIB

GPIB is another option to communicate with instruments. Unfortunately, most GPIB adapters (USB, PCI, PCIe or otherwise) are rather expensive and open source adapters, at the time of this writing, are unsatisfactory. In addition, many USB GPIB adapters are not cross platform, and in many cases are supported only under elderly versions of Microsoft Windows.

Nevertheless, if a GPIB adapter is available, NI VISA and PyVISA greatly ease their usage. To use an instrument connected over GPIB using the PyVISA library, it's as simple as `visa.instrument(GPIB0::12)`.

Appendix C

Anduino Micro-controller Prototyping Platform

C.1 Introduction

In electro-optical design, it is often desirable to have a platform to quickly prototype micro-controller designs. Historically, the Arduino platform has been an excellent example of this, with successful in a wide range of projects and professions [5].

During the course project, an Arduino compatible micro-controller prototyping environment, dubbed the "Anduino", was developed. The Anduino is electrically and mechanically compatible with Arduino shields, enabling rapid development of sensors and illumination sources. The Anduino is single sided, and designed to be compatible with the design rules of the Roland Modella (10 mil spacing and traces) to allow in-house production.

Similar to many of the other circuit boards in this document, the traces on the Anduino are milled with a 0.4 mm square-shouldered, plunge-cutting endmill. The holes, slots and outline are cut with a 0.8 mm square-shouldered, plunge-cutting endmill.

C.2 Design

The Anduino is electrically similar to the Sparkfun Arduino Pro Mini and inspired by Ed Baafi's Fabkit. It carries an Atmel ATMega368, a power and general purpose LED, footprints for high and low power voltage regulators, an 8 or 16 MHz resonator, a USB Mini B or two 0.1 inch headers for power, a FTDI serial header, and standard Arduino compatible 0.1 inch headers. The electrical connections of the 0.1 inch headers are fully electrically compatible (with the small exception of the A_{ref} pin) with a standard Arduino. On prototypes, a voltage regulator was typically not installed, instead the Anduino was powered from a standard USB charger or a USB port on a computer.

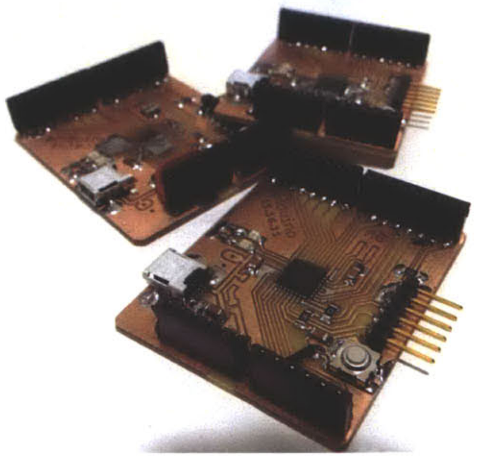


Figure C-1: Generations of early Anduinos.

Though in many ways the Anduino is more flexible than a standard Arduino, some concessions had to be made to accommodate the manufacturing process. The most glaring omission is the lack of an In Circuit Serial Programming header (ICSP). Thankfully, the ICSP header is typically only required once, to flash the bootloader, and a small adapter (discussed below) was fabricated to allow programming. Another glaring omission is the lack of a USB interface. The Anduino requires the use of a USB to TTL serial converter like the FTDI USB-TTL. In addition, the reset pin on the Arduino 0.1 inch headers requires a short jumper wire to connect. On many prototypes, this jumper was omitted because reset is rarely required by external circuitry. Finally, the A_{ref} pin for the ATMega368 analog-to-digital converter (ADC) is not connected to the 0.1 inch headers, and there is no provision to do so (even with a jumper wire). This is an exceedingly rare connection, and was never required during prototyping.

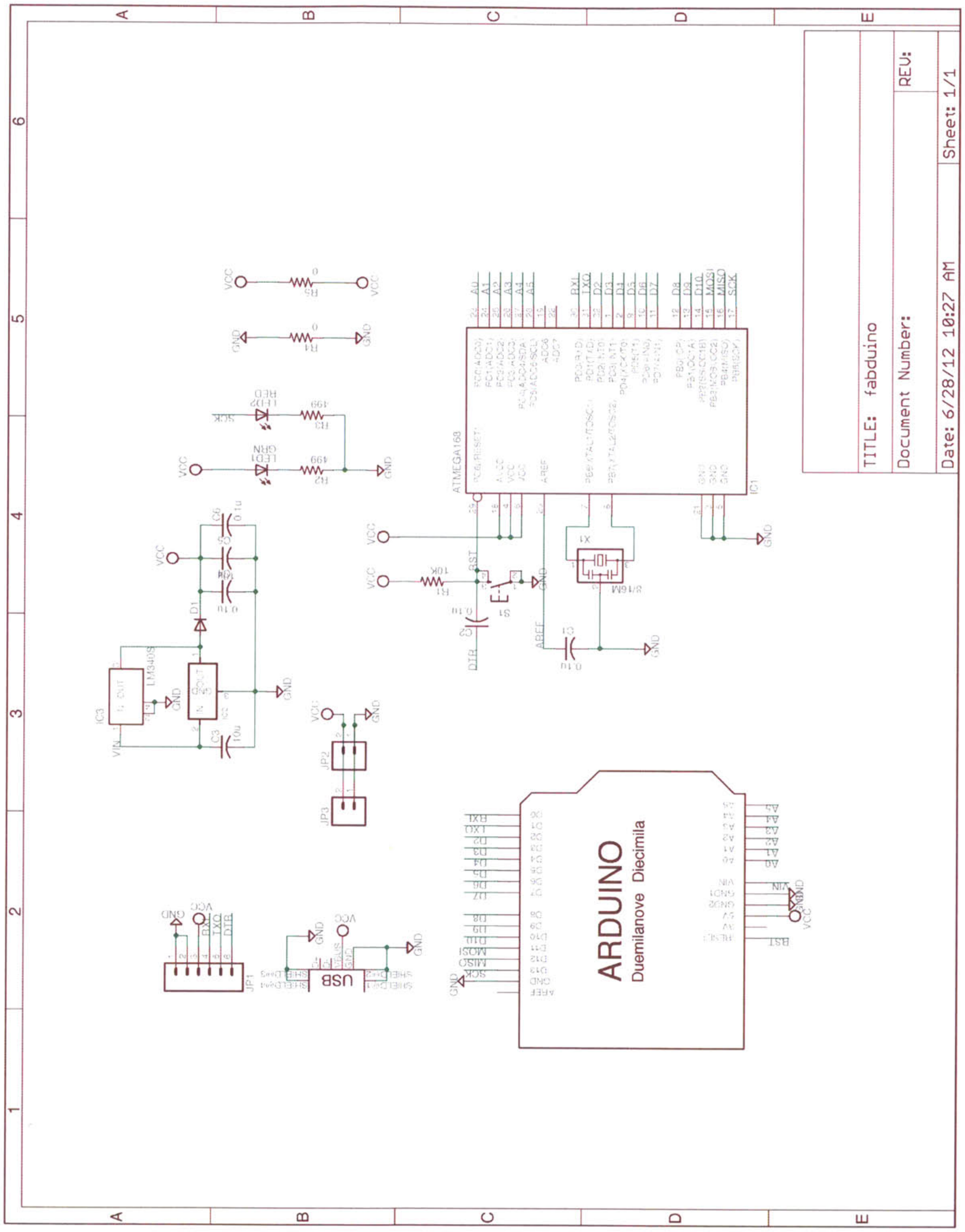


Figure C-2: Anduino schematic.

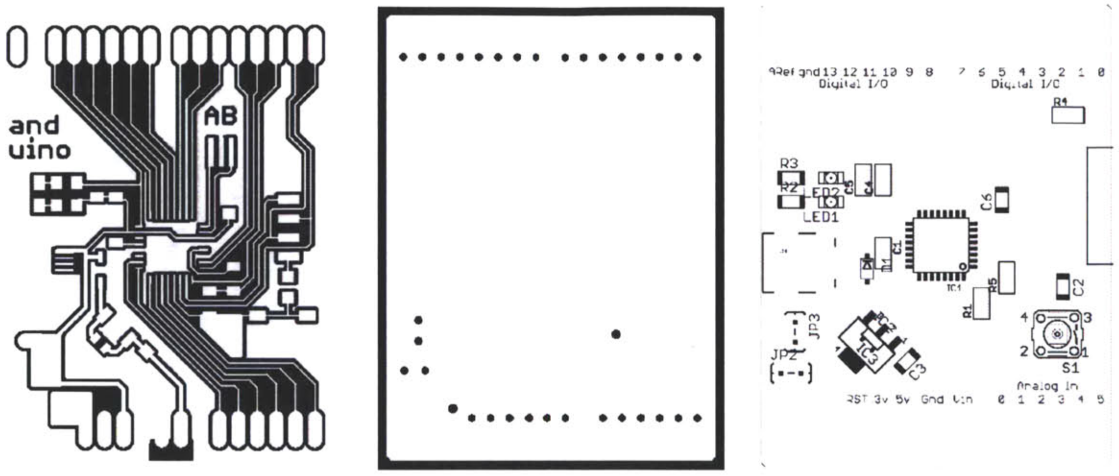
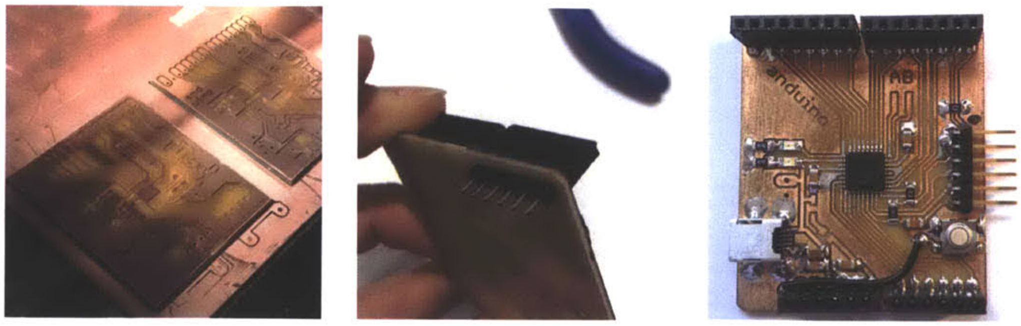

(a) Top layer traces and pads. (b) Holes and board outline. (c) Component placement.

Figure C-3: Diagram of traces, cutout and holes and component placement.

The schematic for the Anduino is shown in Fig C-2. Of particular note are options for either a high or low current output voltage regulator. In addition, there are several methods of powering the board, voltage can be supplied either via the USB Mini B header, JP2, JP3, or VIN on the 0.1 inch Arduino compatible headers. There are two zero ohm jumpers to carry power and ground due to routing limitations on the single-sided board.

The layout of the Anduino is illustrated in Fig C-3. The Anduino circuit board was drawn to accommodate the design rules for the Roland Modella with a 0.4 mm square-shouldered, plunge-cutting endmill. In this configuration, the machine can comfortably hold tolerances of 10 mil traces with 10 mil spacing between traces. Slots, holes and the board outline is cut with a 0.8 mm square-shouldered, plunge-cutting endmill. Consequently, the minimum whole size is 0.8 mm. The boards, shortly after milling on the Roland Modella are shown in Fig C-4a.

The top layer traces and pads, shown in Fig C-3a, reveal a layout similar to a traditional Arduino. The centrally located ATMega368 is surrounded by Arduino compatible headers on the periphery. The FTDI serial header is on the left and the USB Mini B and power entry connectors are on the right above the voltage regulators and bypass capacitance. The power and general purpose LEDs are located below the text marking the board as an Anduino, above the USB header. If the internal RC oscillator is not used, there are provisions for an 8 or 16 MHz resonator.


(a) Two Anduinos shortly after milling on the Roland Modela.

(b) Male headers used to fixture female housings while soldering.

(c) Fully populated Anduino.

Figure C-4: Fabricating the Anduino. Milling the board, populating the headers. Completed Anduino shown on the far right.

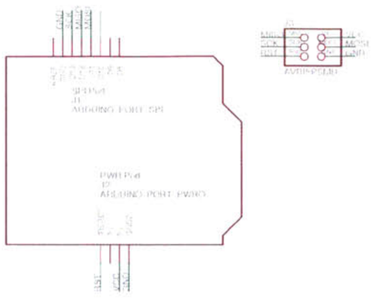
The Arduino headers have elongated pads to facilitate numerous connection and mounting configurations. In particular, they are designed to have right angle headers surface mounted vertically as shown in Fig C-4b.

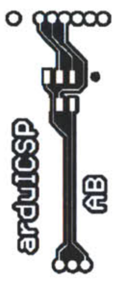
C.3 Loading the Bootloader

To use the Arduino programming environment, the STK500 serial boot-loader must first be programmed onto the ATMega368. Unfortunately, one of the largest omissions on the Anduino is the lack of a standard In Circuit Serial Programming header. Thankfully, it is straightforward to construct an adapter to allow programming through the standard Arduino header pins. The schematic, the top layer traces layout, and a photo of a populated completed board are shown in Fig C-5 below. On the top layer, the dot next to the ICSP header denotes pin one.

C.4 FabISPkey

The Arduino environment can use a number of commercial programmers to load the bootloader onto the AVR ATMega368 micro-controller. In this case, a programmer was developed for this application. This programmer, the FabISPkey is based on Neil Gershenfeld's FabISP which is based on David Mel-


(a) Schematic of ICSP adapter.



(b) Layout.

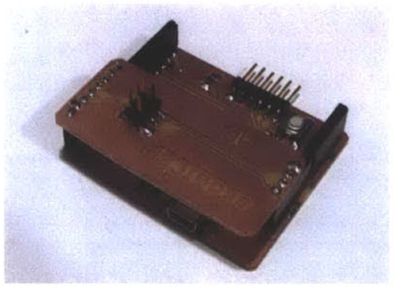

(c) ICSP adapter installed just prior to programming.

Figure C-5: Schematic, layout, and completed and populated ICSP adapter PCB.

lis's FabISP which is in turn based on Limor Fried's USBTinyISP which is based on Dick Streefland's USBTiny [15, 17, 28, 43].

Electrically, the design is a natural evolution of its parent designs. The most striking change is the board edge USB A plug connector. This change greatly informs the form factor of the programmer. In particular, the programmer is now a "key" allowing simpler use. The total size of the programmer has been also greatly reduced making it one of the smallest AVR programmers available. The programmer was typically fabricated out of 1.6 mm thick single sided copper-clad FR1. In many USB ports, the programmer was a little wiggly, affixing a 2.5 mm shim (a piece of a Charlie Card) greatly alleviated this.

When developing a programmer there is an interesting "chicken and egg problem" whereby the programmer must be programmed before use. Since the design is electrically akin to Neil and David's programmer, the bootstrapping process is similar. First, the reset jumper (the solder jumper closest to the processor) should be shorted with a blob of solder. Next, connect the programmer (via the ICSP header) to another (previously bootstrapped) programmer. Then, with the programmer to-be powered, flash its firmware. Power down the programmers, disconnect them, and desolder the solder jumper.

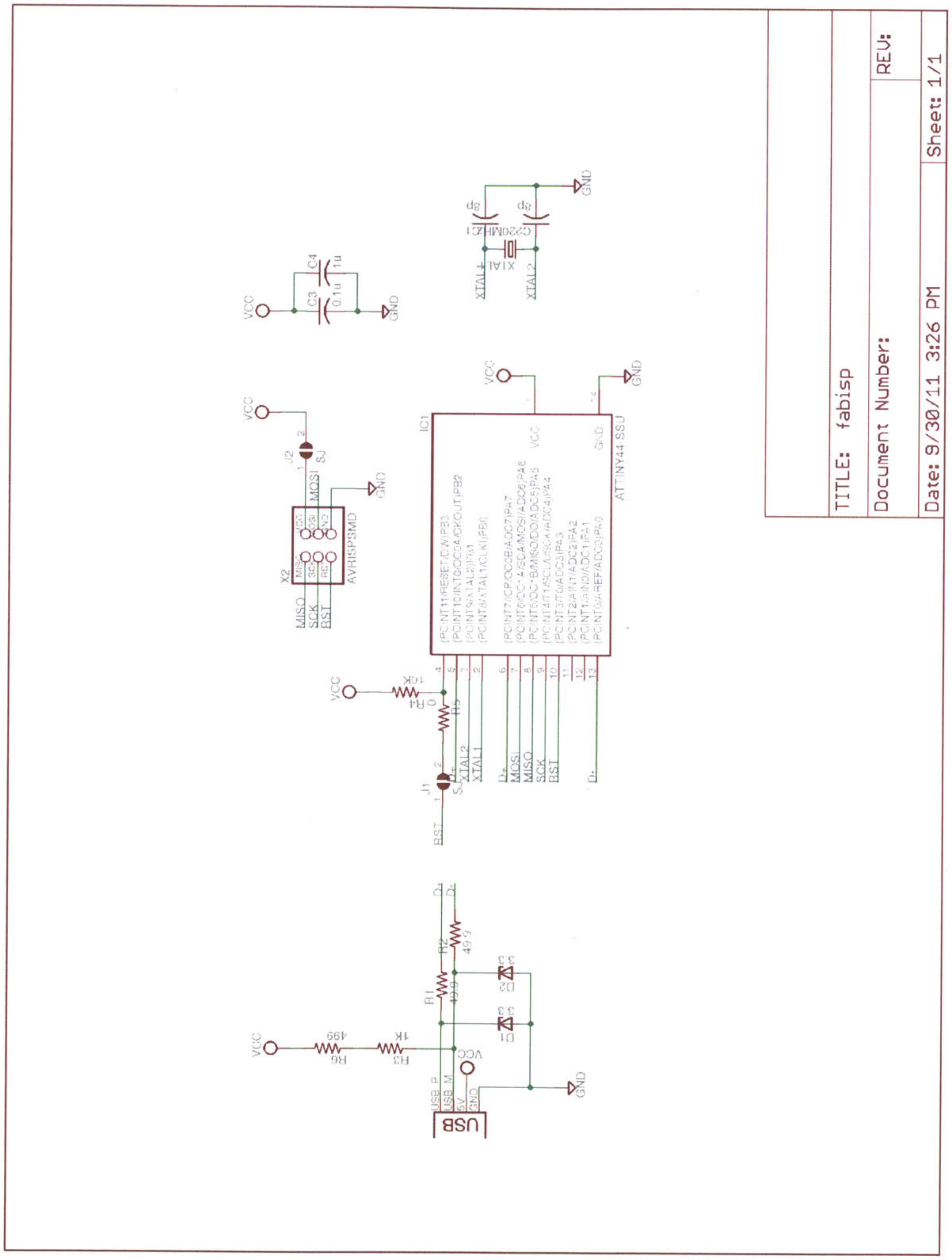
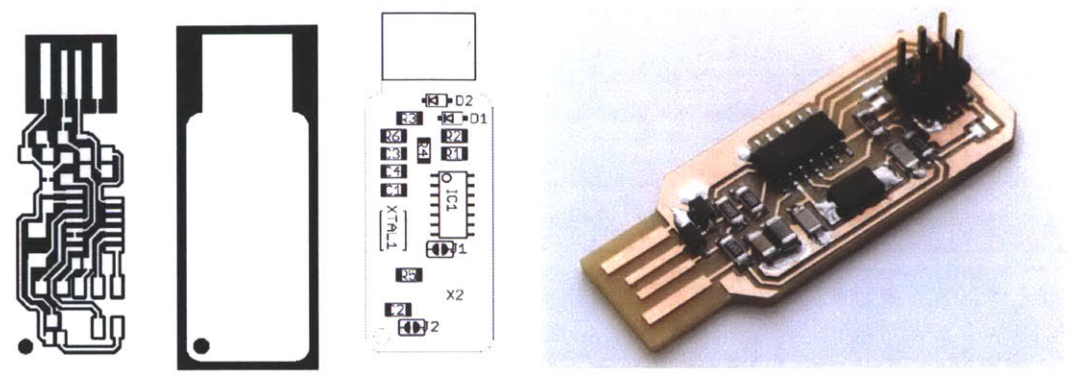


Figure C-6: FabISPkey schematic.



(a) Layout. (b) Outline. (c) Placement diagram. (d) Completed and populated FabISPkey.

Figure C-7: Traces, outline, component placement, and completed and populated FabISPkey.

The schematic for the programmer is illustrated in Fig C-6. The programmer is built around an AT-Tiny44 which uses the V-USB library to communicate to the computer [33]. The circuit, inspired by notes in the V-USB specifications, relies upon zener diodes to lower the logic levels from 5 V to 3.3 V. 50 Ω series termination resistors are used to match the impedance of the USB signal. There is a 1.5 kΩ resistor to force the USB protocol into the slowest speed mode "low speed". A 20 MHz crystal is used to accurately time the USB communications. The most current designs and firmware can be downloaded from the public GIT repository `git://git.bardagjy.com/fabisp`.

Appendix D

Photodiode Amplifier Supplementary Materials

D.1 General Purpose Photodiode Prototype

To rapidly interface varied types of detectors to the existing development platform discussed in Appendix C the general purpose photodiode platform was developed. Theoretical grounding and background concerning it's construction is discussed in Chapter 4. In particular, these supplementary materials concern designs pictured in Fig 4-5.

D.1.1 Schematic

The schematic pictured below in Fig D-1 is essentially a modified transimpedance amplifier which feeds a non-inverting amplifier configured as a voltage follower. Components marked "DNP" or "Do Not Place" are optional passives excluded in the base design. In particular, capacitors C_3 and C_4 are placed in the event the amplifier is unstable and oscillates. Resistor R_2 is placed to bleed the die capacitance in particularly large photodiodes. Since R_1 is a large value, there are few available resistors to fine tune the system gain. As such, R_4 and R_5 can be tuned with more common parts for a similar result.

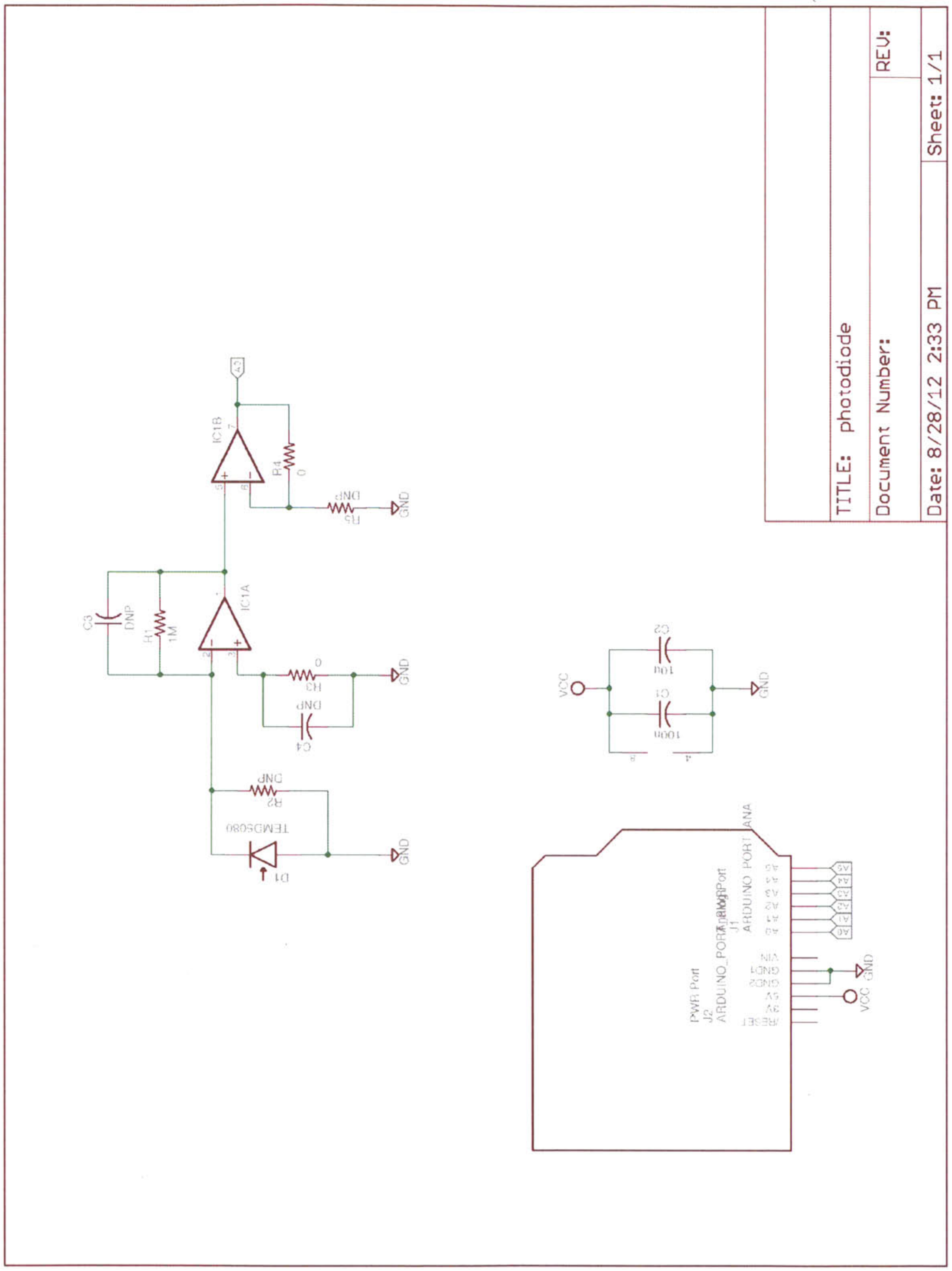
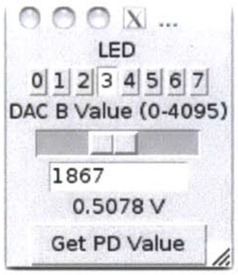


Figure D-1: Extended schematic detailing the general purpose photodiode evaluation board.

D.1.2 Firmware

The firmware used to exercise the general purpose photodiode circuit board is described in Appendix G. The firmware has the ability to select and control the brightness of up to eight LEDs, as discussed in Chapter 5 and Appendix E, and to digitize signals presented to any of its six analog-to-digital converters. The firmware uses a character based command system. When the firmware receives an `l`, it activates the LED numbered by the proceeding byte. The two digital-to-analog converters carried by the illumination shield are adjusted by sending an ASCII `a` or `b` respectively. To read from an analog-to-digital converter, an ASCII `r` is transmitted by the host, and the Arduino responds with the ADC value.

D.1.3 Graphical User Interface



1867
0.5078 V

Figure D-2: Screenshot of graphical user interface used to develop photodiode amplifier.

To aid development, a graphical user interface was developed. From this GUI, the experimenter can select a LED to be illuminated and its control current. When a reading is requested, the GUI displays the voltage presented to its analog-to-digital converted. A screenshot of the GUI is shown Fig D-2.

```
#!/usr/bin/python
#
# gui.py
# Andy Bardagjy
# Aug 29, 2012
#
# This GUI selects and controls the current passed through
# solid state illuminators, once a LED has been lit, the
# photocurrent generated in the photodide can be read and
# displayed in the window
#
```

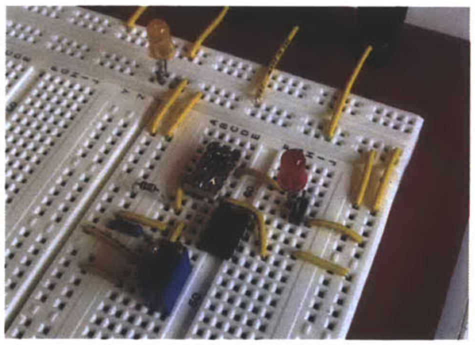
```

from Tkinter import *
import serial
import sys
import time

# GUI root window
root = Tk()
root.wm_title("TLV5638")

# Serial port address via argument
if (len(sys.argv) == 2):
    port = sys.argv[1]
else:
    # Default address
    port = '/dev/tty.usbmodemfa2441'

ser = serial.Serial(port,
                    baudrate=9600,
                    timeout=0.1) # seconds

time.sleep(2)
ser.flush()

# Runs on B scale event
def b_scale_update(x):
    # No Entry "set" method, so delete and insert
    b_entry.delete(0,END)
    b_entry.insert(0,str(b_scale.get()))
    b_set_dac(b_scale.get())

# Runs on B entry event
def b_entry_update(x):
    b_scale.set(int(b_entry.get()))

# Update DAC B
def b_set_dac(val):
    dacAddr = 'b'
    topByte = chr(int(val)/256)
    botByte = chr(val-ord(chr(int(val/256)))*256)
    ser.write(dacAddr + topByte + botByte)

# Runs on LED radiobutton event
def radio_update():
    led_update(v.get())
```

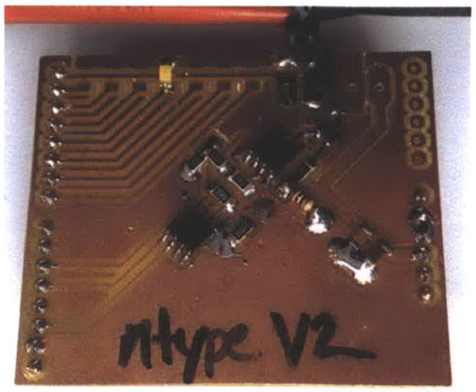
```

# Update the LED
def led_update(val):
    ledAddr = 'l'
    ledVal = chr(int(val))
    ser.write(ledAddr+ledVal)

# Get ADC Value
def get_adc_val():
    # To select ADC by sending 'r'
    sel_adc = 'r'
    # gen_pd use ADC 0
    adc_addr = chr(int(0))
    # Getting an acceptable value out is problematic
    while True:
        try:
            # Select the ADC with sel_adc
            ser.write(sel_adc)
            # Select which ACD to read from
            ser.write(adc_addr)
            # Get the first byte
            b1 = ord(ser.read(size=1))
            # Get the second byte
            b2 = ord(ser.read(size=1))
            # Convert to a voltage
            voltage = round(5.*(b1+b2*256)/1024.,4)
            adc_label_val.set(str(voltage) + " V")
            break
        except TypeError:
            # Wait a bit and try again
            time.sleep(0.1)

# LED Label
led_label = Label(root, text="LED")
led_label.pack(side=TOP)

# LED radio button
btfr = Frame(root)
v = StringVar()
v.set(4)
for led in range(8):
    b = Radiobutton(btfr,
                    text=str(led),
                    variable=v,
                    value=led,
                    indicatoron=0,
                    takefocus=1,
                    command=radio_update)
    b.pack(side=LEFT, anchor=W)
btfr.pack()

# B Label
b_label = Label(root, text="DAC B Value (0-4095)")
b_label.pack()

# B Scale
b_scale = Scale(root,
                from_=0, to=4095,
                orient=HORIZONTAL,
                length=100,
                showvalue=0,
                command=b_scale_update)
b_scale.pack()

# B Entry
b_entry = Entry(root, width=10)
b_entry.pack()
b_entry.bind("<Return>", b_entry_update)

# ADC Value Label
adc_label_val = StringVar()
adc_label_val.set("Voltage")
adc_label = Label(root, textvariable=adc_label_val)
adc_label.pack()

# ADC Read button
adc_button = Button(root, text="Get PD Value", command=get_adc_val)
adc_button.pack()

# run UI
root.mainloop()
```

Appendix E

Programmable Current Source Supplementary Materials

E.1 Introduction


(a) Circuit mocked up on a solder-less breadboard.



(b) Fabricated and populated board.

Figure E-1: From validation prototype to fabricated and populated circuit board.

In semiconductor illuminators the output flux is directly proportional to the current flowing through the device. That is, devices such as light emitting diodes, laser diodes, and super-luminescent diodes are *current controlled*. In the chapter 5, design of several programmable current sources is discussed. In this Appendix, supplementary materials are attached to facilitate reproduction of this work.

E.2 Programmable Current Source Firmware

The firmware below, runs on Arduinos and Arduino clones such as the Anduino discussed in Appendix C. The firmware interprets serial packets, typically sent by a personal computer, to produce a controlled output current. The firmware communicates with "Arduino shields" carrying the Texas Instruments TLV5638 dual 12-bit Digital-to-Analog Converter (DAC) connected via the Serial Peripheral Interface (SPI). The analog voltage produced by the DAC is then converted to a current through a variety of schemes as discussed in Chapter 5.

```
// fw.pde
// Andy Bardagjy
// July 22, 2012
//
// This firmware exercises the ntype constant current test shield
// It sets the DAC, a TI TLV5638 to a voltage which corresponds to
// an outputted current
//
// The ref voltage is set to Vcc / 2 so the full scale of the DAC
// is 5V. This is then divided by 10, so full scale is 500mV.
// The sense resistor is 10 ohm, theoretical full scale
// current output is 50mA
//

#include <SPI.h>

// Pins
const int slaveSelectPin = 10;

// Variables
byte incomingByte = 0;
int set = 0;
int dacVal = 0; // DAC A = 0, DAC B = 1, LED select = 2

void setup() {

    // Initialize the serial port
    Serial.begin(9600);

    // Initialize SPI
    SPI.begin();
    SPI.setDataMode((1 << CPOL) | (0 << CPHA));
    SPI.setClockDivider(SPI_CLOCK_DIV16);

    // Set chip select to OUT
    pinMode(slaveSelectPin, OUTPUT);

    // Set LED pins to OUT and pull it high
    DDRD |= 0xFC; // Port D except serial, Arduino pins 2-7
    DDRB |= 0x03; // Port B, Arduino pins 8,9

    // Turn off all LEDs by default
    PORTD &= ~0xFC; // Pull Port D low, except serial pins (0,1)
    PORTB &= ~0x03; // Pull pins 8,9 low
}

void dacwrite(int chan0, int chan1) {

    // Drop the top nibble
    chan1 = chan1 & 0x0FFF;
    chan0 = chan0 & 0x0FFF;

    // Write packet to DAC B
    // Assert CS by toggling low
    digitalWrite(slaveSelectPin, LOW);

    unsigned int dac1 = 0x1000; //Write value to DAC B Buffer
    unsigned int pkt = dac1 | chan1;

    // Send the first byte
    SPI.transfer((byte)(pkt >> 8));

    // Send the second byte
    SPI.transfer((byte)(pkt & 0x00FF));

    // De-assert CS by toggling CS high
    digitalWrite(slaveSelectPin, HIGH);

    // Write packet to DAC A
    // Assert CS by toggling low
    digitalWrite(slaveSelectPin, LOW);

    unsigned int dac0 = 0x8000; // Write to DAC A and B
    unsigned int npkt = dac0 | chan0;

    // Send the first byte to the DAC
    SPI.transfer((byte)(npkt >> 8));

    // Send the second byte to the DAC
    SPI.transfer((byte)(npkt & 0x00FF));

    // De-assert CS by toggling CS high
    digitalWrite(slaveSelectPin, HIGH);
}

void loop() {

    unsigned int chanA = 0x0000;
      unsigned int chanB = 0x0000;

    if (false) {
        // Set up the DAC, slow, internal ref
        for (int i = 0; i < 3; i++) {
            delay(500);
            SPI.transfer((byte)0x90);
            SPI.transfer((byte)0x02);
        }
    }

    while(1) {
        if (Serial.available() > 0) {
            // Read incoming byte
            incomingByte = Serial.read();

            // if the incoming byte is ASCI a
            if (incomingByte == 97 && set == 0) {
                dacVal = 0;
                set = 2;
            }

            // else if the incoming byte is ASCI b
            else if (incomingByte == 98 && set == 0) {
                dacVal = 1;
                set = 2;
            }

            // else if the incoming byte is ASCI l
        else if (incomingByte == 108 && set == 0) {
            dacVal = 2;
            set = 2;
        }

        // else, modify dac A
        else if (dacVal == 0 && set != 0) {
            if (set == 2) {
                chanA = 0x0000;
                chanA = incomingByte << 8; // MSB
                set--;
            }
            else {
                chanA = chanA | incomingByte; // LSB
                dacwrite(chanA, chanB);
                set--;
            }
        }

        // else, modify dac B
        else if (dacVal == 1 && set != 0) {
            if (set == 2) {
                chanB = 0x0000;
                chanB = incomingByte << 8; // MSB
                set--;
            }
            else {
                chanB = chanB | incomingByte; // LSB
                dacwrite(chanA, chanB);
                set--;
            }
        }

        // else, modify which LED is lit
        else if (dacVal == 2 && set != 0) {

            // First turn off all LEDs
            PORTD &= ~0xFC; // Pull Port D low, except serial pins (0,1)
            PORTB &= ~0x03; // Pull pins 8,9 low

            switch(incomingByte){
                case 0:
                    // Turn on LED 0 connected to pin 2
                    PORTD |= 0x04;
                    break;
                case 1:
                    // Turn on LED 1 connected to pin 3
                    PORTD |= 0x08;
                    break;
                case 2:
                    // Turn on LED 2 connected to pin 4
                    PORTD |= 0x10;
                    break;
                case 3:
                    // Turn on LED 3 connected to pin 5
                    PORTD |= 0x20;
                    break;
                case 4:
                    // Turn on LED 4 connected to pin 6
                    PORTD |= 0x40;
                    break;
                case 5:
                    // Turn on LED 5 connected to pin 7
                    PORTD |= 0x80;
                    break;
                case 6:
                    // Turn on LED 6 connected to pin 8
                    PORTB |= 0x01;
                    break;
                case 7:
                    // Turn on LED 7 connected to pin 9
                    PORTB |= 0x02;
                    break;
                case 8:
                    // Turn off all LEDs
                    PORTD &= ~0xFC; // Pull Port D low, except serial pins (0,1)
                    PORTB &= ~0x03; // Pull pins 8,9 low
            }
            set = 0;
        }
    }
  }
}
```

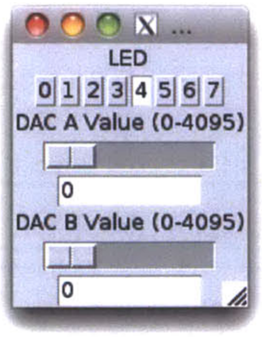


Figure E-2: Screenshot of the control GUI.

E.3 Python GUI for Current Source

The source for the pictured Python GUI is duplicated below and a screenshot of the resulting GUI is reproduced above. This software, presents the experimenter with a control GUI which communicates serially with the Arduino firmware (above), and subsequently the digital-to-analog converter and its voltage to current circuitry as discussed in Chapter 5.

```
#!/usr/bin/python
#
# gui.py
# Andy Bardagjy
# 7/25/2012
#

from Tkinter import *
import serial
import sys

# GUI root window
root = Tk()
root.wm_title("TLV5638")

# Serial port address via argument
if (len(sys.argv) == 2):
    port = sys.argv[1]
else:
    # Default address
    port = '/dev/tty.usbmodemfa2441'

```

```
ser = serial.Serial(port,9600)

# Runs on A scale event
def a_scale_update(x):
    # No Entry "set" method, so delete and insert
    a_entry.delete(0,END)
    a_entry.insert(0,str(a_scale.get()))
    a_set_dac(a_scale.get())

# Runs on A entry event
def a_entry_update(x):
    a_scale.set(int(a_entry.get()))

# Update DAC A
def a_set_dac(val):
    dacAddr = 'a'
    topByte = chr(int(val)/256)
    botByte = chr(val-ord(chr(int(val/256)))*256)
    ser.write(dacAddr + topByte + botByte)

# Runs on B scale event
def b_scale_update(x):
    # No Entry "set" method, so delete and insert
    b_entry.delete(0,END)
    b_entry.insert(0,str(b_scale.get()))
    b_set_dac(b_scale.get())

# Runs on B entry event
def b_entry_update(x):
    b_scale.set(int(b_entry.get()))

# Update DAC B
def b_set_dac(val):
    dacAddr = 'b'
    topByte = chr(int(val)/256)
    botByte = chr(val-ord(chr(int(val/256)))*256)
    ser.write(dacAddr + topByte + botByte)

# Runs on LED radiobutton event
def radio_update():
    led_update(v.get())

def led_update(val):
    ledAddr = 'l'
    ledVal = chr(int(val))
    ser.write(ledAddr+ledVal)

# LED Label
led_label = Label(root, text="LED")
led_label.pack(side=TOP)

# LED radio button
btfr = Frame(root)
v = StringVar()
v.set(4)
for led in range(8):
    b = Radiobutton(btfr,
                    text=str(led),
                    variable=v,
                    value=led,
                    indicatoron=0,
                    takefocus=1,
                    command=radio_update)
    b.pack(side=LEFT, anchor=W)
btfr.pack()

# A Label
a_label = Label(root, text="DAC A Value (0-4095)")
a_label.pack()

# A Scale
a_scale = Scale(root,
                from_=0, to=4095,
                orient=HORIZONTAL,
                length=100,
                showvalue=0,
                command=a_scale_update)
a_scale.pack()

# A Entry
a_entry = Entry(root, width=10)
a_entry.pack()
a_entry.bind("<Return>", a_entry_update)

# B Label
b_label = Label(root, text="DAC B Value (0-4095)")
b_label.pack()

# B Scale
b_scale = Scale(root,
                     from_=0, to=4095,
                     orient=HORIZONTAL,
                     length=100,
                     showvalue=0,
                     command=b_scale_update)
b_scale.pack()

# B Entry
b_entry = Entry(root, width=10)
b_entry.pack()
b_entry.bind("<Return>", b_entry_update)

# run UI
root.mainloop()
```

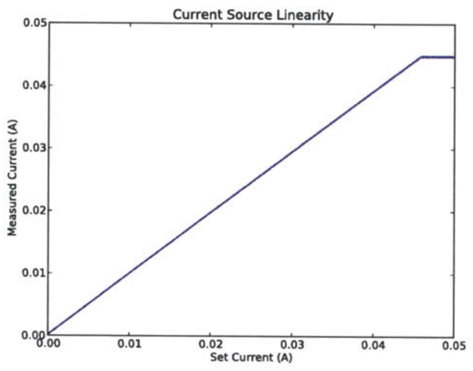
E.4 Current Source Evaluation

The following software evaluates the linearity of the programmable current source.

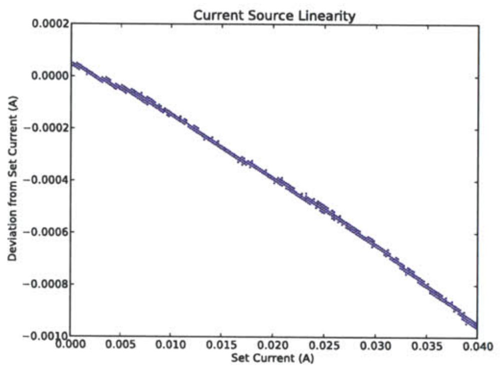
```
#!/usr/bin/env arch -i386 python
#
# lin.py
# Andy Bardagjy
# July 27,2012
#
# PyVISA calls NI visa which is compiled with 32 bits,
# so python run in 32 bit with #!/usr/bin/env arch -i386 python
#
# The Agilent 34401A is controlled via serial
# Use NI-VISA Configuration to find which ASRL it is connected to
#

from visa import *
import time
import csv
import serial

# CSV open file
f = open('out.csv', 'w')
writer = csv.writer(f)
```

```

# Open the serial port
ser = serial.Serial('/dev/tty.usbmodemfa2441',9600)

# Agilent 34401A
dmm_addr = 'ASRL1'
dmm = SerialInstrument(dmm_addr,
                        baud_rate = 9600,
                        data_bits = 7,
                        stop_bits = 2,
                        parity = even_parity,
                        term_chars = CR+LF,
                        end_input = term_chars_end_input,
                        send_end = True,
                        delay = 1
                        )

# Set it up for remote access, reset to factory, remote again
print "Resetting DMM"
dmm.write('SYST:REM')
dmm.write('*RST')
time.sleep(20)
dmm.write('SYST:REM')

print dmm_addr + ": " + dmm.ask('*IDN?')

# Loop from 0 to 6 volts in 10mV increments cyc times
cyc = 15

sets = map(lambda x: x*16, range(257))

writer.writerow(sets)

for I in range(cyc):
    tarr = []
    for J in range(len(sets)):
        setpoint = sets[J]
        # Set the current source to the setpoint
        ser.write('b' +
                  chr(int(setpoint)/256) +
                  chr(setpoint-ord(chr(int(setpoint/256)))*256) )
        # Let the output settle 1 seconds
        time.sleep(1)
        # Measure the output
        meas = dmm.ask('MEAS:CURR:DC? 100 MA, MAX')
        tarr.append(meas)
        # Tell the user what's up
        print("Cycle: " + str(I) +
              " set:" + str(setpoint) +
              "meas:" + str(meas))
    # Write tarr to the file
    writer.writerow(tarr)

# Close the file
f.close()
```

The following software takes the above measurements and produces the resulting plots pictured below



(a) Measured current on set current of the circuit.



(b) Zoomed portion of measured linearity of circuit. The circuit is very linear.

Figure E-3: Experimental results of Fig 5-8 linearity.

The plot on the left, shows measured current on DAC code. The measured current produced appears to be rather linear for much of the range of the device. The maximum current the source can sink is around 45 mA. On the right, the plot illustrates the measured error between the predicted source current and the measured current through the device. The error is roughly linear indicating there may be a systematic error in the measurement, device or both. That said, the maximum error is less than 1 mA, which is acceptable for its purposes.

```
#!/usr/bin/python
# plot.py
# Andy Bardagjy
# 7/31/2012
#
# Grabs stored data csv from lin.py and plots it
```

```
#

import csv
from pylab import *

f = open('out.csv', 'r')
reader = csv.reader(f)
cols = zip(*reader) # transpose reader so it's columnwise

# Figure and axis
fig = figure()
ax = fig.add_subplot(111)

# Titles and labels
ax.set_title("Current Source Linearity")
ax.set_xlabel("Set Current (A)")
ax.set_ylabel("Deviation from Set Current (A)")

# Loop through the set and measured currents columnwise
for col in cols[0:-1]:
    # Plot the measured values on the set values
    ax.plot((float(col[0])/81920)*ones(len(col[1:])),
            map(lambda y: float(y) - float(col[0])/81920, col[1:]),
            'b,',mec='b')

# Close the CSV file
f.close()

# Set axis limits
ax.set_xlim([0,0.05])

# Save the figure
#fig.savefig('lin.pdf')

# Display the figure
show()
```

Appendix F

Illuminator Characterization Supplementary Materials

F.1 Characterization Routine

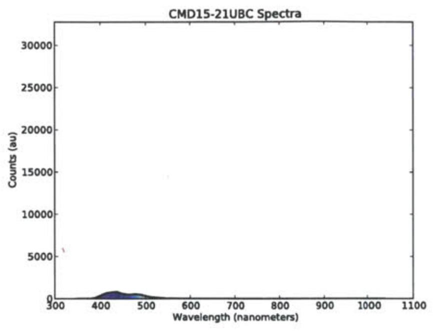
This code measures the electro-optical properties of LEDs using the test fixture described in Chapter 7. To measure the spectral properties of the illuminators, an Ocean Optics USB4000 VIS-NIR fiber attached USB spectrometer [34]. Light from the illuminators on the text fixture is coupled into the spectrometer with a jacketed 400 μm optical fiber.

The following application first establishes communication with the spectrometer and test fixture. Communication with the USB4000 is thanks to a third party interface library developed by Tamás Haraszti [34]. Since the test fixture is running firmware (Appendix E) developed for the constant current test fixtures (Chapter 5), interfacing is via straightforward serial communication.

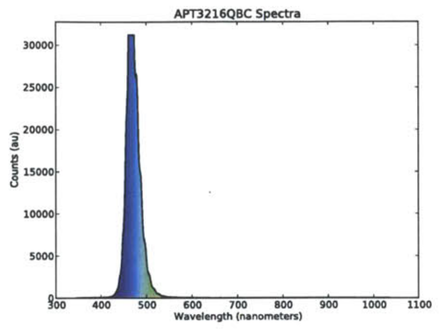
Once communication has been established, the user is prompted via the command line to select which device to illuminate and a filename (typically the part number of the illuminator) to save the figure, and data collected from the device. To reduce stray light contaminants, the tests were run in a light-tight enclosure, and dark frame subtraction was employed. Once the user has inputted a device number and file name, the selected LED is illuminated. The application source code is reproduced below.

```
#!/usr/bin/python
#
# spec.py
# Andy Bardagjy
# August 19, 2012
#
# Creates plots using a Ocean Optics USB4000 spectrometer
#

import OceanOptics
from wavelen2rgb import wavelen2rgb

import numpy as np
from pylab import *
import serial
import time
import csv

# Number of readings the spectrometer averages over
READINGS = 5

# This sets the current through the device
def b_set_dac(val):
    dacAddr = 'b'
    topByte = chr(int(val)/256)
    botByte = chr(val-ord(chr(int(val/256)))*256)
    ser.write(dacAddr + topByte + botByte)

# This selects the LED
def led_update(val):
    ledAddr = 'l'
    ledVal = chr(int(val))
    ser.write(ledAddr+ledVal)

# Converts wavelength to color for plotting
# Respects wavelengths outside visible light
def ptcolor(wavelen):
    if wavelen > 780:
        wavelen = 780
    elif wavelen < 380:
        wavelen = 380
    color = wavelen2rgb(wavelen,MaxIntensity=255)
    color = map(lambda x: x/255.0, color)
    return(color)

```

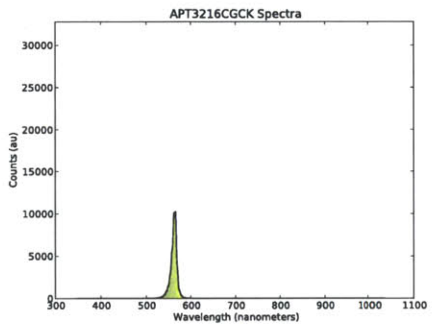
```
# Connect to the spectrometer
while True:
    try:
        OceanOptics.openUSB()
        break
    except IOError:
        print("Spectrometer not found, is it connected?")

# Get the caibration/lookup values from the spectrometer
cal = np.asarray(OceanOptics.GetParameters())

#the detector has this many points
x = np.arange(3670-22)

#get the wavelength scale
x = cal[0] + x*(cal[1] + x*(cal[2]+ x*cal[3]))

# Connect to the test fixture
port = '/dev/tty.usbmodemfa231'
while True:
    try:
        ser = serial.Serial(port,9600)
        break
    except serial.serialutil.SerialException:
        print("Fixture serial port " + port + " not found")
        port = str(raw_input("Enter serial port: "))

# Main loop
while True:

    # Main "menu"
    inpt = str(raw_input("Type LED number [0-7] or e[X]it: "))

    # Exit the program
    if inpt in 'xXqQ':
        break

    elif inpt in '01234567':
        fname = str(raw_input("Enter the filename (part number): "))

        # Turn OFF the LED, takes a few cycles
        for I in range(4):
            b_set_dac(0)
            led_update(int(8))
            time.sleep(1)

        # Grab a dark frame
        dark = np.asarray(OceanOptics.GetSpectra(READINGS)[22:3670])

        # Turn ON the LED, takes a few cycles
        for I in range(4):
            b_set_dac(100)
            led_update(int(inpt))
            time.sleep(1)

        # Get spectra with LED on
        y = np.asarray(OceanOptics.GetSpectra(READINGS)[22:3670])

        # Perform dark correction
        for I in range(len(y)):
            y[I] = y[I] - dark[I]

        # Plotting stuff
        fig = figure()
        ax = fig.add_subplot(111)
        ax.set_title(fname + " Spectra")
        ax.set_xlabel("Wavelength (nanometers)")
        ax.set_ylabel("Counts (au)")
        ylim([0,32768]) # 2^15
        # Color points according to prceived color of wavelength
        for I in range(len(x)-1):
            fill_between(x[I:I+2],y[I:I+2],zeros(2),
                 color=ptcolor(x[I+1]))
            plot(x[I:I+2],y[I:I+2],
                 #color=ptcolor(x[I+1]),
                 color='black',
                 linestyle='solid',
                 linewidth=1.1)

        # Save the figure
        fig.savefig('imgs/' + fname + '.pdf')

        # Save the raw data
        fle = open('data/' + fname + '.csv', 'w')
        writer = csv.writer(fle)
        writer.writerow(x)
        writer.writerow(y)
        fle.close()
```



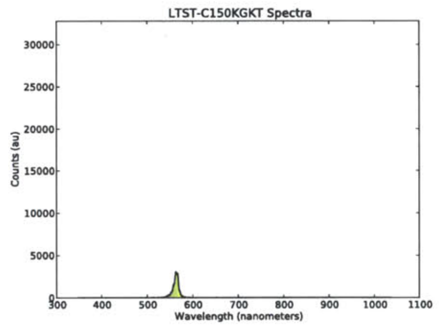
(a) CMD15-21UBC.



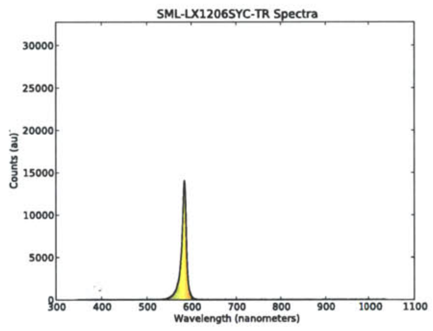
(b) APT3216QBC.



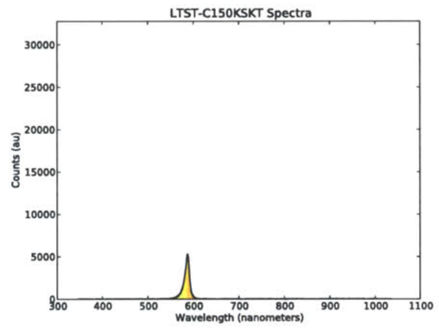
(c) APT3216CGCK.



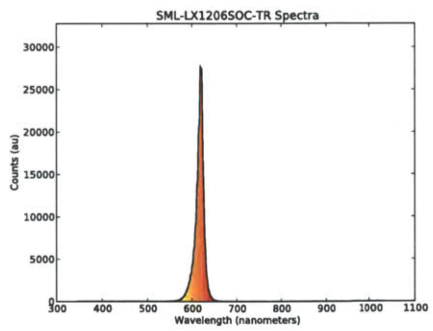
(d) LTST-C150KGKT.



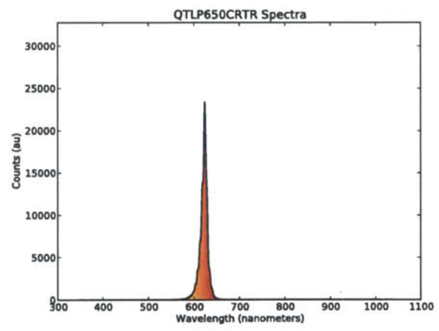
(e) SML-LX1206SYC-TR.



(f) LTST-C150KSKT.

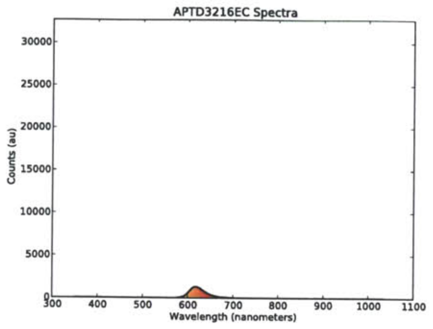


(g) SML-LX1206SOC-TR.

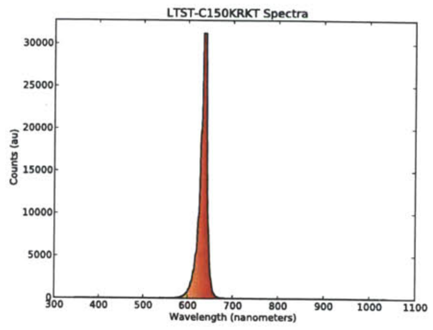


(h) QTLP650CRTR.

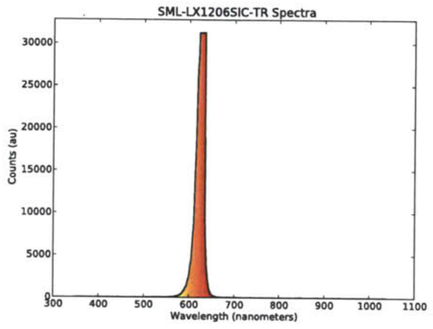
Figure F-1: Fixture 1 LED spectra. LEDs range from 420 nm to 624 nm center wavelength.



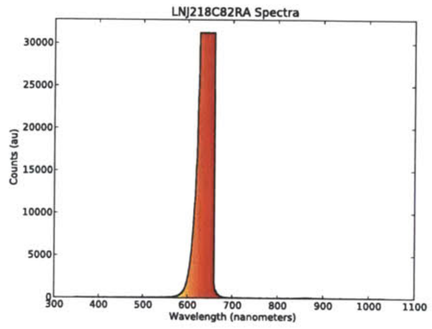
(a) APTD3216EC.



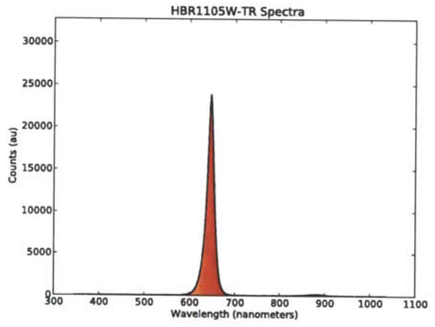
(b) LTST-C150KRKT.



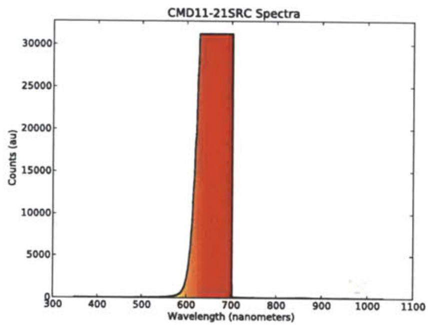
(c) SML-LX1206SIC-TR.



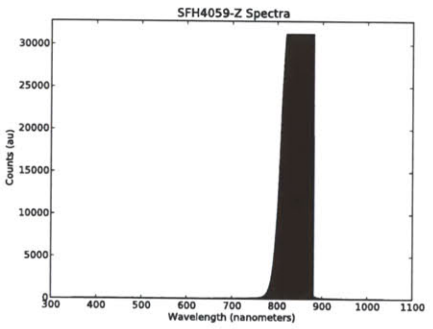
(d) LNJ218C82RA.



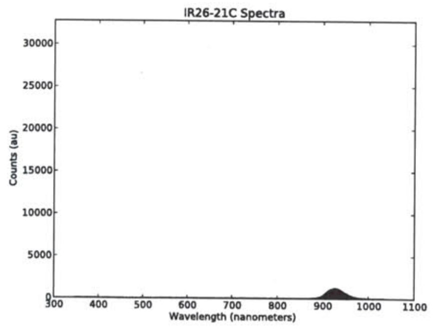
(e) HBR1105W-TR.



(f) CMD11-21SRC.



(g) SFH 4059-Z.



(h) IR26-21C.

Figure F-2: Fixture 2 LED spectra. LEDs range from 625 nm to 940 nm center wavelength.

In the plots, each illuminator is driven with an equal current represented with the digital-to-analog code 100. By the equation described in 5.10, a code of 100 corresponds to.

$$\frac{C}{10 \times 2^{13}} = 1.2\,\mathrm{mA} \tag{F.1}$$

F.2 Normalized Illuminator Characterization

Similar software was developed to explore the normalized spectral emission of the illuminators. This software (reproduced below) searched through the drive currents until an approximately equal maximum output brightness was achieved.

Since the search through drive currents can be a time consuming process, some user interaction niceties were added to facilitate long term data collection. The program warns the user when to swap the spectrometer input fiber to the correct device, and prompts the user to enter the part number of the device under test.

F.2.1 Adaptive Current Spectral Characterization

```python
#!/usr/bin/python
#
# spec.py
# Andy Bardagjy
# August 21, 2012
#
# Searches through LED currents so we can find normalized spectral values
# makes plots using a Ocean Optics USB4000 spectrometer
#

import OceanOptics
from wavelen2rgb import wavelen2rgb

import numpy as np
from pylab import *
import serial
import time
import csv
import os

# Number of readings the spectrometer averages over
READINGS = 5

# This sets the current through the device
def b_set_dac(val):
    dacAddr = 'b'
    topByte = chr(int(val)/256)
    botByte = chr(val-ord(chr(int(val/256)))*256)
    ser.write(dacAddr + topByte + botByte)

# This selects the LED
def led_update(val):
    ledAddr = 'l'
    ledVal = chr(int(val))
    ser.write(ledAddr+ledVal)

# Converts wavelength to color for plotting
# Respects wavelengths outside visible light
def ptcolor(wavelen):
    if wavelen > 780:
        wavelen = 780
    elif wavelen < 380:
        wavelen = 380
    color = wavelen2rgb(wavelen,MaxIntensity=255)
    color = map(lambda x: x/255.0, color)
    return(color)

# Connect to the spectrometer
while True:
    try:
        OceanOptics.openUSB()
        break
    except IOError:
        print("Spectrometer not found, is it connected?")

# Get the caibration/lookup values from the spectrometer
cal = np.asarray(OceanOptics.GetParameters())

#the detector has this many points
x = np.arange(3670-22)

#get the wavelength scale
x = cal[0] + x*(cal[1] + x*(cal[2]+ x*cal[3]))
```

```

# Connect to the test fixture
port = '/dev/tty.usbmodemfa231'
while True:
    try:
        ser = serial.Serial(port,9600)
        break
    except serial.serialutil.SerialException:
        print("Fixture serial port " + port + " not found")
        port = str(raw_input("Enter serial port: "))

# Main loop
while True:

    # Notify user that input is required
    os.system('say user input requested')

    # Main "menu"
    inpt = str(raw_input("Type LED number [0-7] or e[X]it: "))

    # Exit the program
    if inpt in 'xXqQ':
        break

    elif inpt in '01234567':
        fname = str(raw_input("Enter the filename (part number): "))

        # Search through DAC values to find maximum current without
        # saturating the spectrometer
        # Start at DAC code 50 search to 300 in steps of 10
        # Max spectrometer value 2^15
        # search till max is greater than X, less than 2^15
        for dac_val in np.arange(16,4072,16):
            print("dac_val " + str(dac_val))
            # Turn ON the LED, takes a few cycles
            for I in range(4):
                b_set_dac(dac_val)
                led_update(int(inpt))
                time.sleep(1)

            # Get spectra with LED on
            y = np.asarray(OceanOptics.GetSpectra(READINGS)[22:3670])

            # Test if the brightness is within spec
            if y.max() > (2**15 - 2**13) and y.max() < 2**15:
                break

        # Print the dac_val selected
        print("")
        print("SELECTED dac_val: " + str(dac_val))
        print(str(y.max()))
        print("LED number: " + str(inpt))
        print("File name: " + str(fname))
        print("")

        # Dark frame subtraction
        # Turn OFF the LED, takes a few cycles
        for I in range(4):
            b_set_dac(0)
            led_update(int(8))
            time.sleep(1)

        # Grab a dark frame
        dark = np.asarray(OceanOptics.GetSpectra(READINGS)[22:3670])

        # Perform dark correction
        for I in range(len(y)):
            y[I] = y[I] - dark[I]

        # Set up the plot
        fig = figure()
        ax = fig.add_subplot(111)
        ax.set_title(fname + " Spectra at DAC code " + str(dac_val))
        ax.set_xlabel("Wavelength (nanometers)")
        ax.set_ylabel("Counts (au)")
        ylim([0,32768]) # 2^15
        # Color points according to prceived color of wavelength
        for I in range(len(x)-1):
            fill_between(x[I:I+2],y[I:I+2],zeros(2),
                 color=ptcolor(x[I+1]))
            plot(x[I:I+2],y[I:I+2],
                 #color=ptcolor(x[I+1]),
                 color='black',
                 linestyle='solid',
                 linewidth=1.1)

        # Save the figure
        fig.savefig('imgs/dac_' + str(dac_val) + '_' + fname + '.pdf')

        # Save the raw data
        fle = open('data/dac_' + str(dac_val) + '_' + fname + '.csv', 'w')
        writer = csv.writer(fle)
        writer.writerow(x)
        writer.writerow(y)
        fle.close()
```

F.2.2 Normalized Spectral Emission Visualization

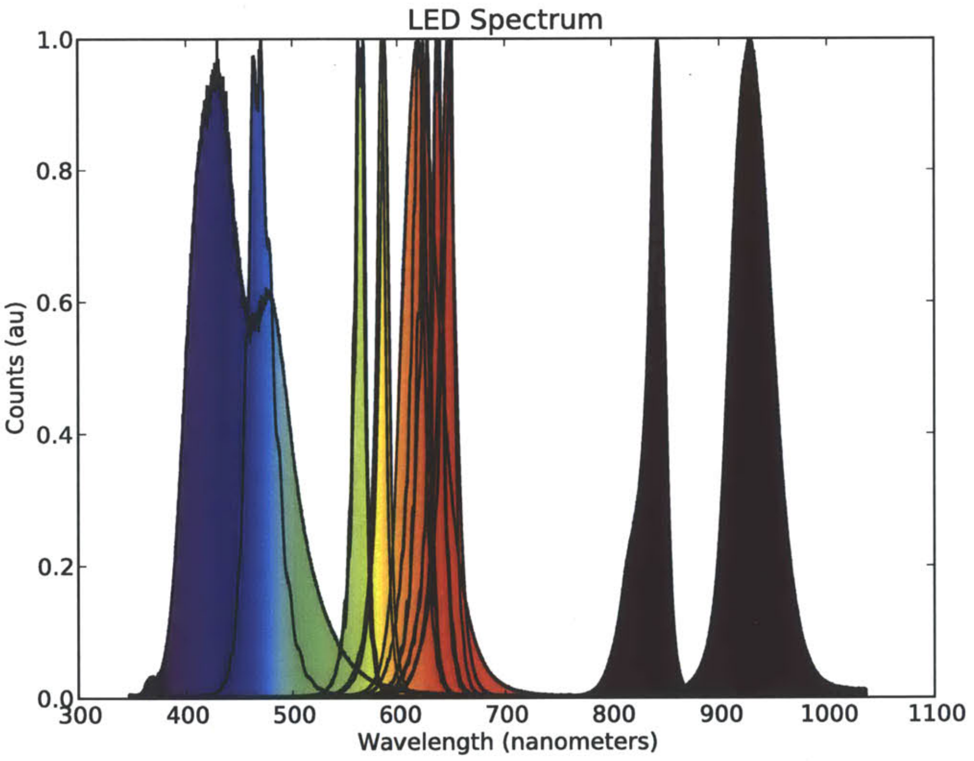


Figure F-3: LED spectra normalized and plotted together.

Once each illuminator was evaluated, the results were compiled using the following program to produce a plot of their normalized spectral characteristics. The combined spectra of the illuminators evaluated are illustrated in Fig F-3 above.

```
#!/usr/bin/python
#
```

```
# comp_plot.py
# Andy Bardagjy
# August 21, 2012
#
# Creates plots using a Ocean Optics USB4000 spectrometer
#

import numpy as np
from pylab import *
import csv
import os
from wavelen2rgb import wavelen2rgb

# Converts wavelength to color for plotting
# Respects wavelengths outside visible light
def ptcolor(wavelen):
    if wavelen > 780:
        wavelen = 780
    elif wavelen < 380:
        wavelen = 380
    color = wavelen2rgb(wavelen,MaxIntensity=255)
    color = map(lambda x: x/255.0, color)
    return(color)

# Directory containing the csv data files
datadir = 'data/'
while True:
    try:
        fnames = os.listdir(datadir)
        break
    except OSError:
        print("Data directory " + datadir + " not found")
        datadir = str(raw_input("Please enter data directory: "))

# Set up the plot
fig = figure()
ax = fig.add_subplot(111)
ax.set_title("LED Spectrum")
ax.set_xlabel("Wavelength (nanometers)")
ax.set_ylabel("Counts (au)")
ylim([0, 1])

for fname in fnames:
    fname = datadir + fname
    fhandle = open(fname,'r')
    reader = csv.reader(fhandle)
    xx = reader.next()
    yy = reader.next()
    fhandle.close()

    # Normalize the readings
    y = np.array(yy,dtype=float)
    x = np.array(xx,dtype=float)
    y = y-y.min()
    y = y/y.max()

    # Color points according to prceived color of wavelength
    for I in range(len(x)-1):
        fill_between(x[I:I+2],y[I:I+2],zeros(2),
             color=ptcolor(float(x[I+1])))
        plot(x[I:I+2],y[I:I+2],
             #color=ptcolor(x[I+1]),
             color='black',
             linestyle='solid',
             linewidth=1.1)

fig.savefig('comp_plot.pdf')
```

Appendix G

Design *&* Prototypes Supplementary Materials

G.1 Unsupervised Transmission Spectroscopy

G.1.1 Illuminator Circuit Board

The illuminator circuit board is designed similarly to the programmable current sources discussed in Chapter 5 and Appendix E. The illuminator circuit board is designed to electrically interface with an Arduino as a "shield". In this work these shields were often used in conjunction with an Arduino compatible clone, the Anduino, discussed in Appendix C. The circuit, reproduced in Fig G-1, uses a transistor buffered transconductance amplifier to convert voltage from a 12-bit DAC to a programmable current. The circuit can supply from zero to 40 mA to one of eight 1206-packaged light emitting diodes.

The circuit was constructed in-house similarly to the other circuits discussed in this work. A copper clad FR1 substrate is milled using a 0.40 mm end mill to cut the traces, and a 0.80 mm end mill to cut through-holes and the board outline. The circuit was designed such that the smallest traces are no less than 10 mils and the smallest spacing between traces is no less than 10 mils.

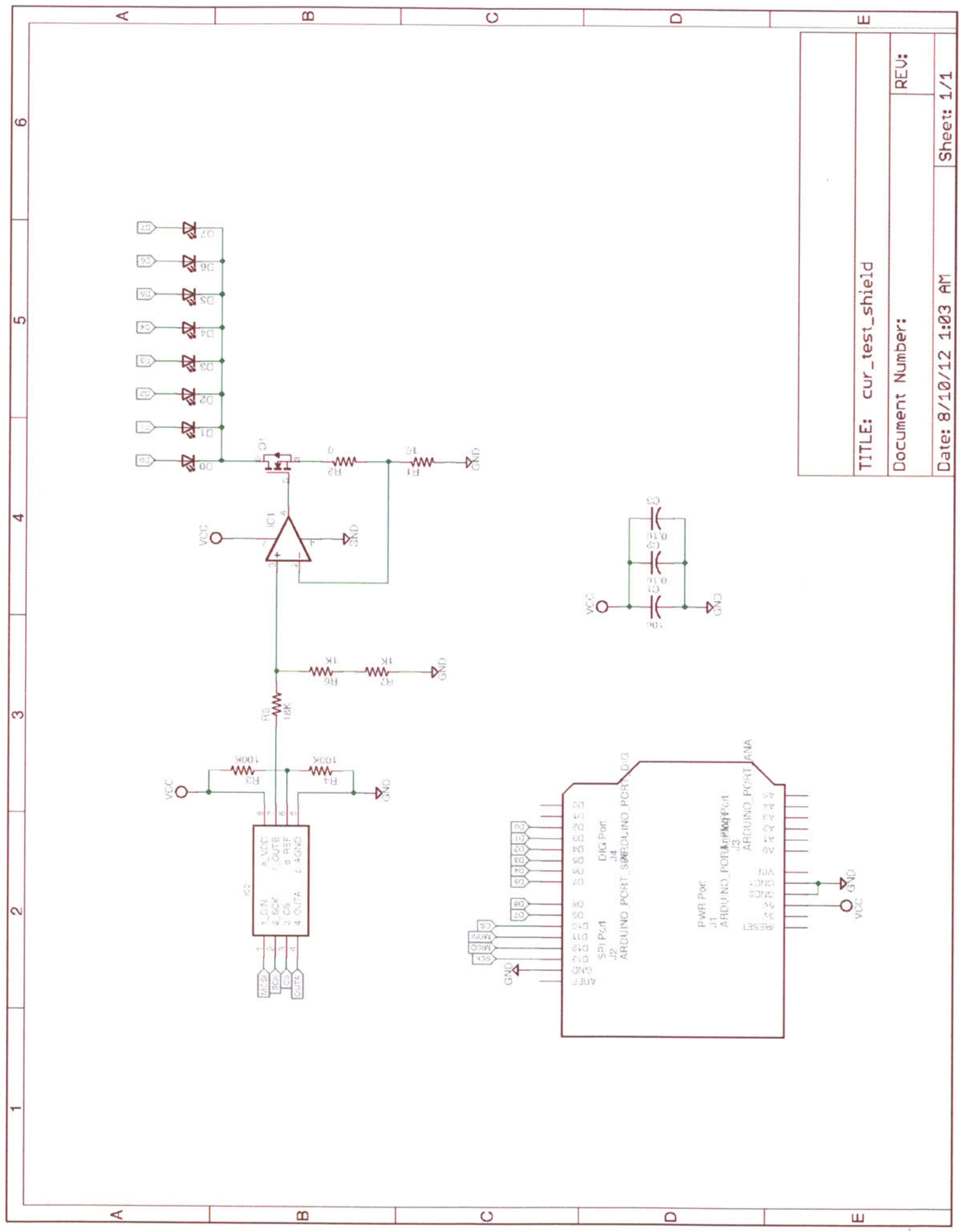
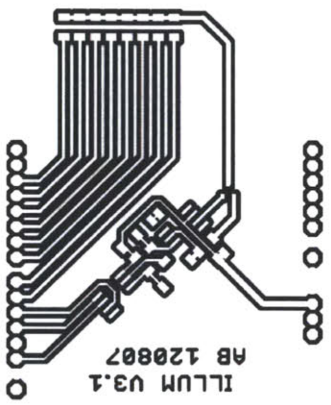
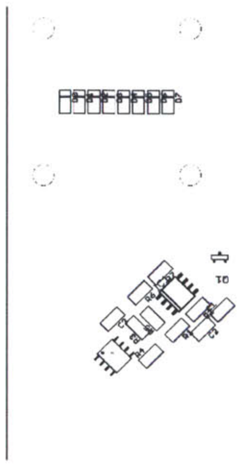


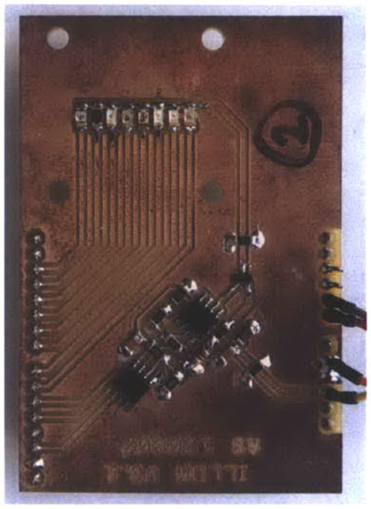
Figure G-1: Illuminator circuit board schematic.



(a) Top layer traces and pads.



(b) Component placement diagram.



(c) Fabricated and populated illuminator. In just a few days, the copper is heavily oxidized.

Figure G-2: Top layer board artwork, placement diagram, and fabricated and populated circuit board.

In Fig G-2a, the top layer trace artwork is reproduced. Dark regions denote copper to be removed. In Fig G-2b, the component placements are specified. Finally, in Fig G-2c, the fabricated and populated circuit board is shown. The circuit board shows quite a bit of oxidation despite being fabricated only days prior to the photo. The ground plane has been relieved around the power and analog input ports to allow connection of an expansion cable for the detector circuit board.

G.1.2 Detector Circuit Board

The detector circuit board is similar to the photodiode amplifier circuit board discussed in Chapter 4 and Appendix D. The circuit board is designed to be carried as an Arduino shield, though is typically connected via a short umbilical to an illuminator circuit board.

The detector circuit board carries a pair of TEMD5080X01 photodiodes which are each connected to a transimpedance amplifier. The voltage from each transimpedance amplifier is buffered with a unity-gain non-inverting amplifier. The transimpedance amplifier and the voltage follower each use one half of an OPA2340, a dual operational amplifier.

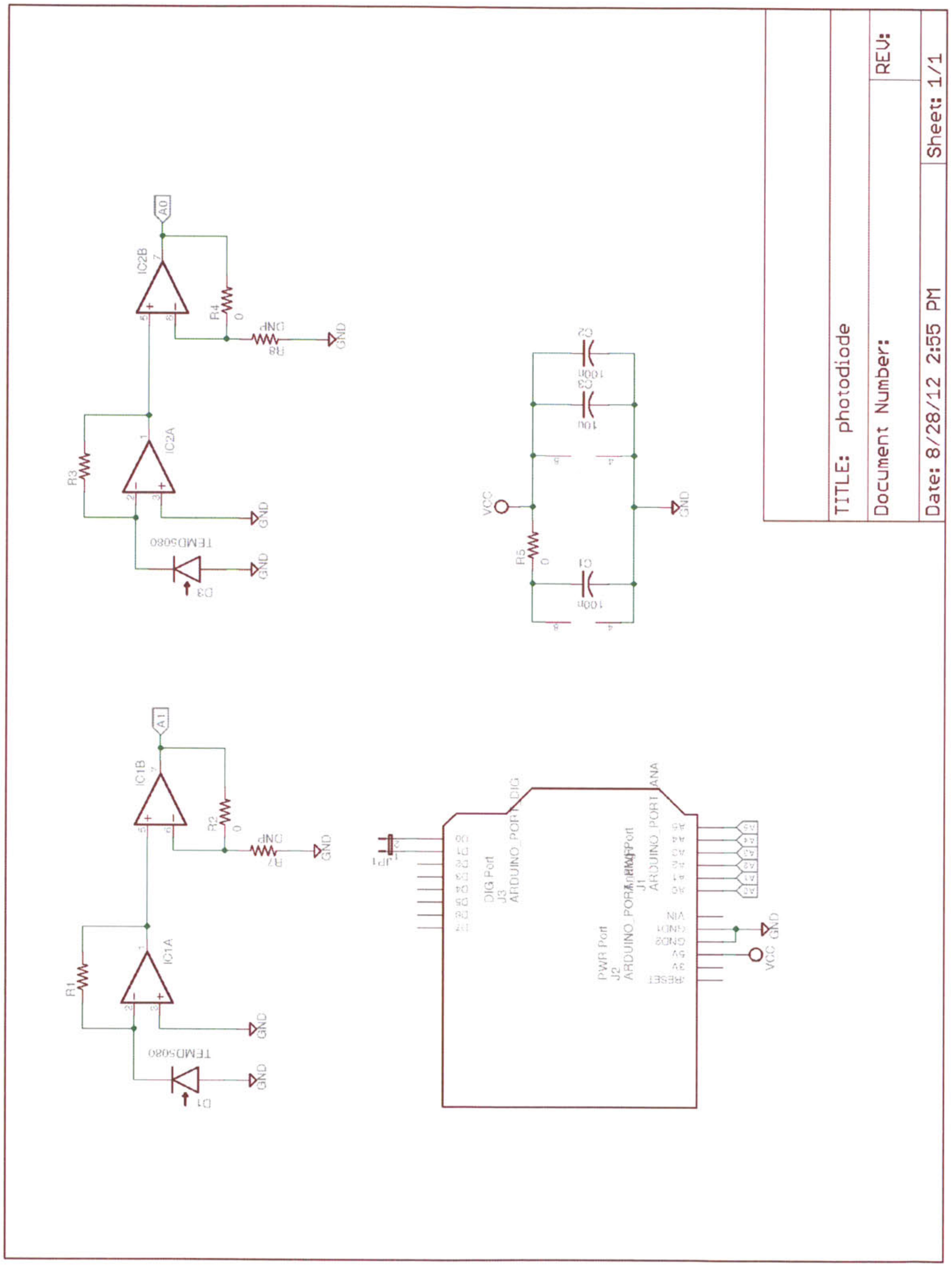
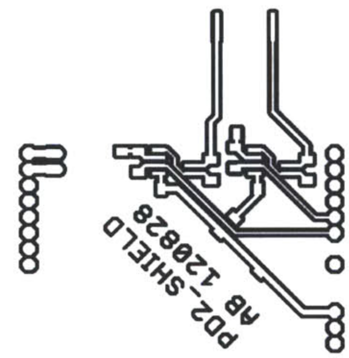
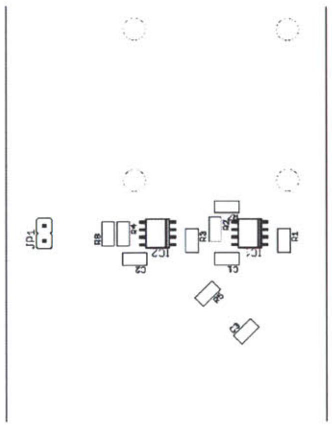


Figure G-3: Photodetector circuit board schematic.


(a) Top layer traces and pads.


(b) Component placement diagram.

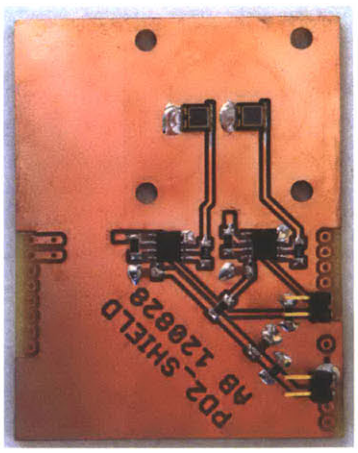

(c) Fabricated and populated detector.

Figure G-4: Top layer board artwork, placement diagram, and fabricated and populated circuit board.

The photodetector circuit board is pictured in Fig G-4. The top layer artwork is pictured in Fig G-4a, the component placement diagram is shown in Fig G-4b, and the completed photodetector circuit board is pictured in Fig G-4c. Similar to the illuminator circuit board, the ground plane has been relieved to accommodate connectors for the wire harness from the illuminator circuit board.

G.1.3 Firmware

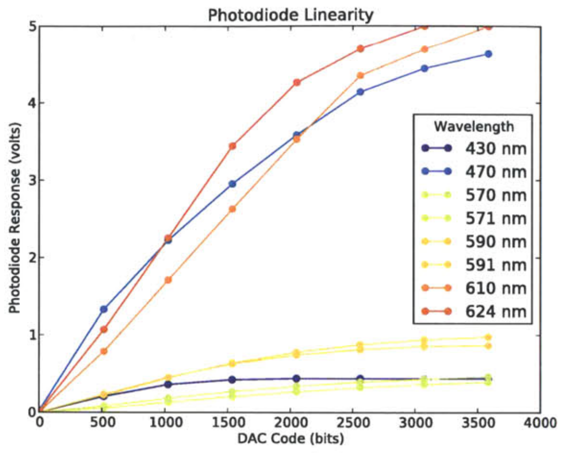
The firmware below has the ability to select and control the brightness of up to eight LEDs, as discussed in Chapter 5 and Appendix E, and to digitize signals presented to any of its six analog-to-digital converters. The firmware uses a character based command system. When the firmware receives an `l`, it activates the LED numbered by the proceeding byte. The two digital-to-analog converters carried by the illumination shield are adjusted by sending an ASCII `a` or `b` respectively. To read from an analog-to-digital converter, an ASCII `r` is transmitted by the host, and the Arduino responds with the ADC value.

```
// fw.ino
// Andy Bardagjy
// Aug 30, 2012
//
// This firmware is an interface for the illum and gen_pd shield
// It sets the DAC, a TI TLV5638 to a voltage which corresponds to
// an outputted current. It also interfaces with the single photodiode
// on the gen_pd shield
//
// The ref voltage is set to Vcc / 2 so the full scale of the DAC
// is 5V. This is then divided by 10, so full scale is 500mV.
// The sense resistor is 10 ohm, theoretical full scale
// current output is 50mA
//

#include <SPI.h>

// Pins
const int slaveSelectPin = 10;

// Variables
byte incomingByte = 0;
int set = 0;
int dacVal = 0; // DAC A = 0, DAC B = 1, LED select = 2, read A0 = 3

void setup() {

    // Initialize the serial port
    Serial.begin(9600);

    // Initialize SPI
    SPI.begin();
    SPI.setDataMode((1 << CPOL) | (0 << CPHA));
    SPI.setClockDivider(SPI_CLOCK_DIV16);

    // Set chip select to OUT
    pinMode(slaveSelectPin, OUTPUT);

    // Set LED pins to OUT and pull it high
    DDRD |= 0xFC; // Port D except serial, Arduino pins 2-7
    DDRB |= 0x03; // Port B, Arduino pins 8,9

    // Turn off all LEDs by default
    PORTD &= ~0xFC; // Pull Port D low, except serial pins (0,1)
    PORTB &= ~0x03; // Pull pins 8,9 low

}

void dacwrite(int chan0, int chan1) {

    // Drop the top nibble
    chan1 = chan1 & 0x0FFF;
    chan0 = chan0 & 0x0FFF;

    // Write packet to DAC B
    // Assert CS by toggling low
    digitalWrite(slaveSelectPin, LOW);

    unsigned int dac1 = 0x1000; //Write value to DAC B Buffer
    unsigned int pkt = dac1 | chan1;

    // Send the first byte
    SPI.transfer((byte)(pkt >> 8));

    // Send the second byte
    SPI.transfer((byte)(pkt & 0x00FF));

    // De-assert CS by toggling CS high
    digitalWrite(slaveSelectPin, HIGH);

    // Write packet to DAC A
    // Assert CS by toggling low
    digitalWrite(slaveSelectPin, LOW);

    unsigned int dac0 = 0x8000; // Write to DAC A and B
    unsigned int npkt = dac0 | chan0;

    // Send the first byte to the DAC
    SPI.transfer((byte)(npkt >> 8));

    // Send the second byte to the DAC
    SPI.transfer((byte)(npkt & 0x00FF));

    // De-assert CS by toggling CS high
    digitalWrite(slaveSelectPin, HIGH);
}

void loop() {

    unsigned int chanA = 0x0000;
      unsigned int chanB = 0x0000;
    unsigned int dacVal = 0x0000;

    if (false) {
        // Set up the DAC, slow, internal ref
        for (int i = 0; i < 3; i++) {
            delay(500);
            SPI.transfer((byte)0x90);
            SPI.transfer((byte)0x02);
        }
    }

    while(1) {
        if (Serial.available() > 0) {
            // Read incoming byte
            incomingByte = Serial.read();

            // if the incoming byte is ASCI a
            if (incomingByte == 97 && set == 0) {
                dacVal = 0;
                set = 2;
            }

            // else if the incoming byte is ASCI b
            else if (incomingByte == 98 && set == 0) {
                dacVal = 1;
                set = 2;
            }

            // else if the incoming byte is ASCI l
            else if (incomingByte == 108 && set == 0) {
                dacVal = 2;
                set = 1;
            }

            // else if the incoming byte is ASCI r
            else if (incomingByte == 114 && set == 0) {
                dacVal = 3;
                set = 1;
            }

            // else, modify dac A
            else if (dacVal == 0 && set != 0) {
                if (set == 2) {
                    chanA = 0x0000;
                    chanA = incomingByte << 8; // MSB
                    set--;
                }
                else {
                    chanA = chanA | incomingByte; // LSB
                    dacwrite(chanA, chanB);
                    set--;
        }
    }

    // else, modify dac B
    else if (dacVal == 1 && set != 0) {
        if (set == 2) {
            chanB = 0x0000;
            chanB = incomingByte << 8; // MSB
            set--;
        }
        else {
            chanB = chanB | incomingByte; // LSB
            dacwrite(chanA, chanB);
            set--;
        }
    }

    // else, modify which LED is lit
    else if (dacVal == 2 && set != 0) {

        // First turn off all LEDs
        PORTD &= ~0xFC; // Pull Port D low, except serial pins (0,1)
        PORTB &= ~0x03; // Pull pins 8,9 low

        switch(incomingByte){
            case 0:
                // Turn on LED 0 connected to pin 2
                PORTD |= 0x04;
                break;
            case 1:
                // Turn on LED 1 connected to pin 3
                PORTD |= 0x08;
                break;
            case 2:
                // Turn on LED 2 connected to pin 4
                PORTD |= 0x10;
                break;
            case 3:
                // Turn on LED 3 connected to pin 5
                PORTD |= 0x20;
                break;
            case 4:
                // Turn on LED 4 connected to pin 6
                PORTD |= 0x40;
                break;
                case 5:
                    // Turn on LED 5 connected to pin 7
                    PORTD |= 0x80;
                    break;
                case 6:
                    // Turn on LED 6 connected to pin 8
                    PORTB |= 0x01;
                    break;
                case 7:
                    // Turn on LED 7 connected to pin 9
                    PORTB |= 0x02;
                    break;
                case 8:
                    // Turn off all LEDs
                    PORTD &= ~0xFC; // Pull Port D low,
                                    // except serial pins (0,1)
                    PORTB &= ~0x03; // Pull pins 8,9 low
            }
            set = 0;
        }

        // else, read analog port
        else if (dacVal == 3 && set !=0) {

            switch(incomingByte){
                case 0:
                    // Read from analog port
                    dacVal = analogRead(A0);
                    // Send one byte at a time
                    Serial.write(dacVal & 0xFF);
                    Serial.write((dacVal >> 8) & 0xFF);
                    break;
                case 1:
                    // Read from analog port
                    dacVal = analogRead(A1);
                    // Send one byte at a time
                    Serial.write(dacVal & 0xFF);
                    Serial.write((dacVal >> 8) & 0xFF);
                    break;
                case 2:
                    // Read from analog port
                    dacVal = analogRead(A2);
                    // Send one byte at a time
                    Serial.write(dacVal & 0xFF);
                    Serial.write((dacVal >> 8) & 0xFF);
```

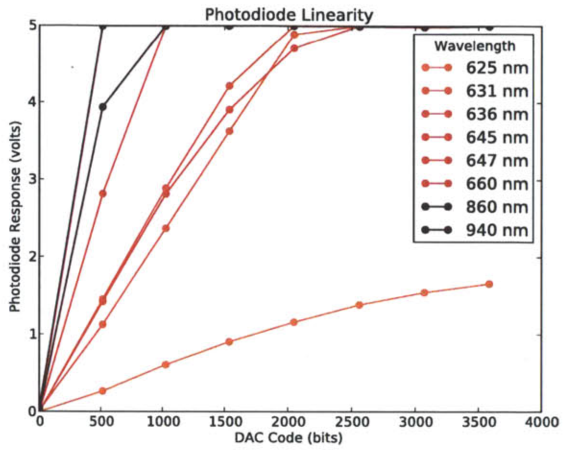
```
                            break;
                        case 3:
                            // Read from analog port
                            dacVal = analogRead(A3);
                            // Send one byte at a time
                            Serial.write(dacVal & 0xFF);
                            Serial.write((dacVal >> 8) & 0xFF);
                            break;
                        case 4:
                            // Read from analog port
                            dacVal = analogRead(A4);
                            // Send one byte at a time
                            Serial.write(dacVal & 0xFF);
                            Serial.write((dacVal >> 8) & 0xFF);
                            break;
                        case 5:
                            // Read from analog port
                            dacVal = analogRead(A5);
                            // Send one byte at a time
                            Serial.write(dacVal & 0xFF);
                            Serial.write((dacVal >> 8) & 0xFF);
                            break;
                    }
                    set = 0;
                }

            }
        }
}
```

G.1.4 Measuring Linearity

Software was developed to evaluate the linearity of the illuminator photodiode system. The program, reproduced below, first prompts the user to select the serial port the fixture is connected to. Next, the user inputs a `csv` file which indicates the hardware configuration of the fixture. The program then scans through a set of increasing diode currents for each device in the fixture's configuration. The photocurrent induced in the photodiode is converted to a voltage and digitized. The response is then plotted. Examples of produced plots are reproduced above in Fig G-5.



(a) Linearity of illuminators and photodiodes on fixture one.



(b) Linearity of illuminators and photodiodes on fixture two.

Figure G-5: Linearity of the illuminator and photodiode system.

```
#!/usr/bin/python
#
# lin.py
# Andy Bardagjy
# August 29, 2012
#
# Measures the linearity of photodiode
#

import numpy as np
from pylab import *
import serial
import time
import sys
import csv
import os
from wavelen2rgb import wavelen2rgb

ADC_NUM = 0

# Converts wavelength to color for plotting
# Respects wavelengths outside visible light
def pltcolor(wavelen):
    if wavelen > 780:
        wavelen = 780
    elif wavelen < 380:
        wavelen = 380
    color = wavelen2rgb(wavelen,MaxIntensity=255)
    color = map(lambda x: x/255.0, color)
    return(color)

# Set the current through the device
def b_set_dac(val):
    dacAddr = 'b'
    topByte = chr(int(val)/256)
    botByte = chr(val-ord(chr(int(val/256)))*256)
    ser.write(dacAddr + topByte + botByte)

# Selects the LED
def led_update(val):
    ledAddr = 'l'
    ledVal = chr(int(val))
    ser.write(ledAddr+ledVal)

# Get ADC Value
def get_adc_val(addr):
    # To select ADC by sending 'r'
    sel_adc = 'r'
    adc_addr = chr(addr)
    # Getting an acceptable value out is problematic :/
    while True:
        try:
            # Select the ADC with sel_adc
            ser.write(sel_adc)
            # Select which ACD to read from
            ser.write(adc_addr)
            # Get the first byte
            b1 = ord(ser.read(size=1))
            # Get the second byte
            b2 = ord(ser.read(size=1))
            # Convert to a voltage
            voltage = round(5.*(b1+b2*256)/1024.,4)
            return voltage
            break
        except TypeError:
            # Wait a bit and try again
            time.sleep(0.1)

# Connect to serial port
port = '/dev/tty.usbmodemfa231'
while True:
    try:
            ser = serial.Serial(port,9600)
            break
        except serial.serialutil.SerialException:
            print("Serial port " + port + " not found")
            port = str(raw_input("Enter serial port: "))

# Set up the figure, plot and axes
fig = figure()
ax = fig.add_subplot(111)
ax.set_title("Photodiode Linearity")
ax.set_xlabel("DAC Code (bits)")
ax.set_ylabel("Photodiode Response (volts)")
ax.set_autoscale_on(True)
ylim([0,5]) # Axis in volts, full scale 5V

# Range of dac values considered
stp = 512
dac_vals = np.arange(0,2**12 - 1,stp)
# Measurements
meas  = []
light = []
dark = []

while True:

    # Warn the user
    os.system('say user input requested')
    inpt = str(raw_input("Enter LED .csv file name or [q]uit: "))

    # Exit
    if inpt in 'qQxX':
        break

    # Otherwise, try to open the file
    try:
        fle = open(inpt,'r')
    except IOError:
        continue

    # Parse the CSV file
    reader = csv.reader(fle)

    # for LED on the test fixture
    for row in reader:

        led_num = int(row[4])
        wavelen = int(row[2])
        part_no = row[1].strip()

        for dac_val in dac_vals:

            # Turn ON the LED, takes a few cycles
            for I in range(4):
                b_set_dac(dac_val)
                led_update(led_num)
                time.sleep(1)

            # Grab an ON photodiode reading
            # gen_pd use ADC 0
            lt = max([get_adc_val(0), get_adc_val(1)])
            light.append(lt)

            # Dark frame subtraction
            # Turn OFF the LED, takes a few cycles
            for I in range(4):
                b_set_dac(0)
                led_update(8)
                time.sleep(1)

            # Grab a dark photodiode reading
            dk = max([get_adc_val(0), get_adc_val(1)])
            dark.append(dk)

            # Perform dark correction
            meas.append(lt-dk)

            # Print the LED info
            print("LED #: " + str(led_num) + " "
                  + part_no + ": " + str(wavelen) + " nm")
            # Print the DAC code
            print("Code: " + str(dac_val) + " Reading: " + str(lt-dk) + " V")

        #  Plot
        ax.plot(dac_vals, meas,
                linestyle='-',
                marker='o',
                color=pltcolor(wavelen),
                markeredgecolor=pltcolor(wavelen),
                label=str(wavelen)+' nm')

        # set everything back to zero
        meas  = []
        light = []
        dark = []

    legend(loc='best', title='Wavelength')

    # Save the figure
    fig.savefig('lin.pdf')

    # Close the file
    fle.close()
```

G.2 Supervised Transmission Spectroscopy

In the case where the sample spectra are well understood, a special purpose instrument is designed for that particular application. To this end, a hand-held special purpose spectrometer platform was developed. This platform is composed of two circuit boards and a mechanical enclosure. The illuminator circuit board contains a micro-controller, indicator LEDs, a pushbutton, and pads for up to six 1206 packaged sample illumination LEDs. The detector circuit board can carry up to two TEMD5080X01 photodiodes. The photodiodes are connected to one half of a OPA2340 dual operational amplifier configured as a transimpedance amplifier. The transimpedance amplifier converts the photocurrent developed in the photodiode to a voltage. This voltage is digitized by the micro-controller on the illuminator circuit board. The micro-controller then analyzes the sensed spectral features to discriminate the sample. The schematics for the illuminator and detector circuit boards are reproduced in the figures below. The board layouts, placement diagrams and photos of the completed board can be found in Chapter 8.

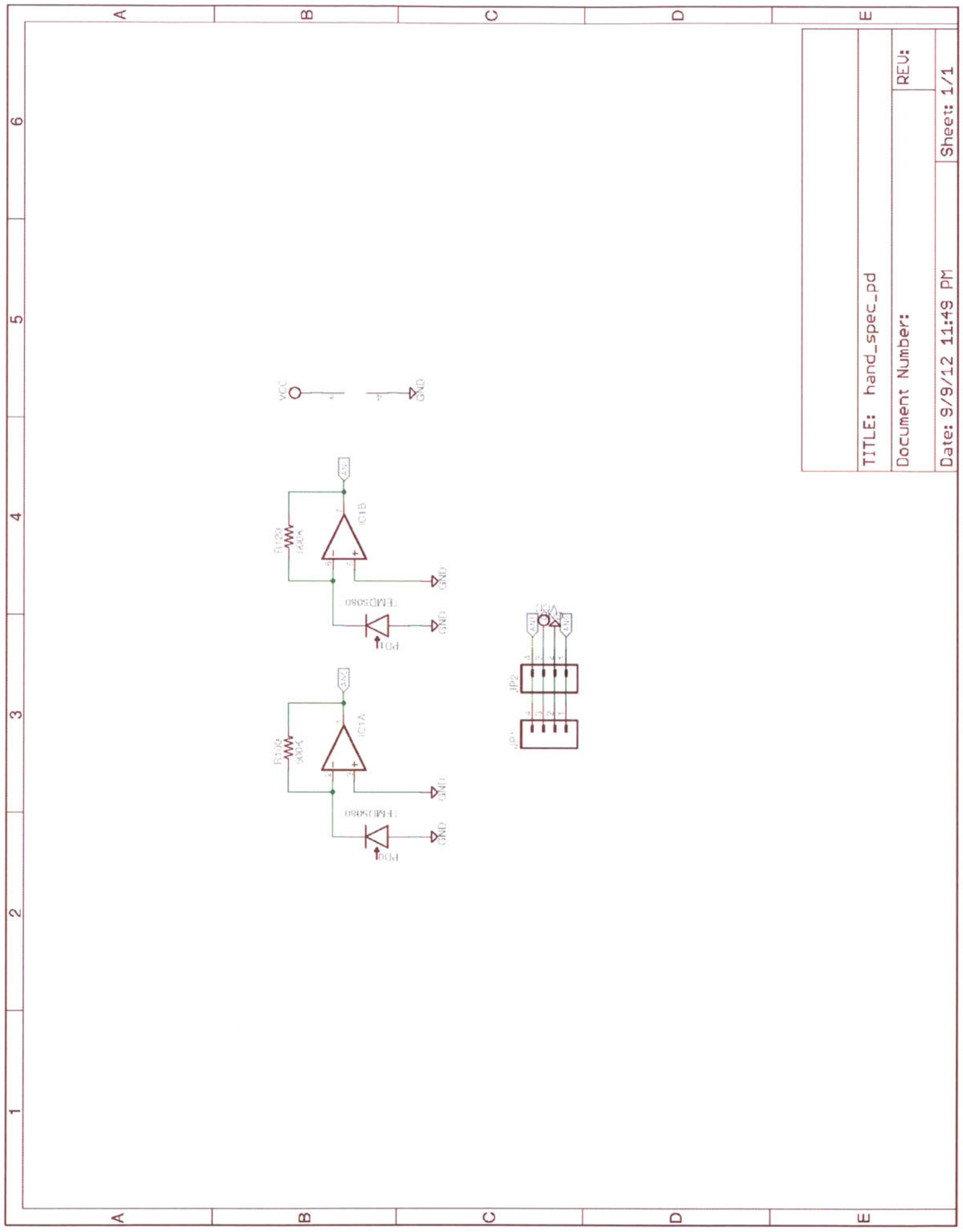


Figure G-6: Photodetector circuit board schematic.

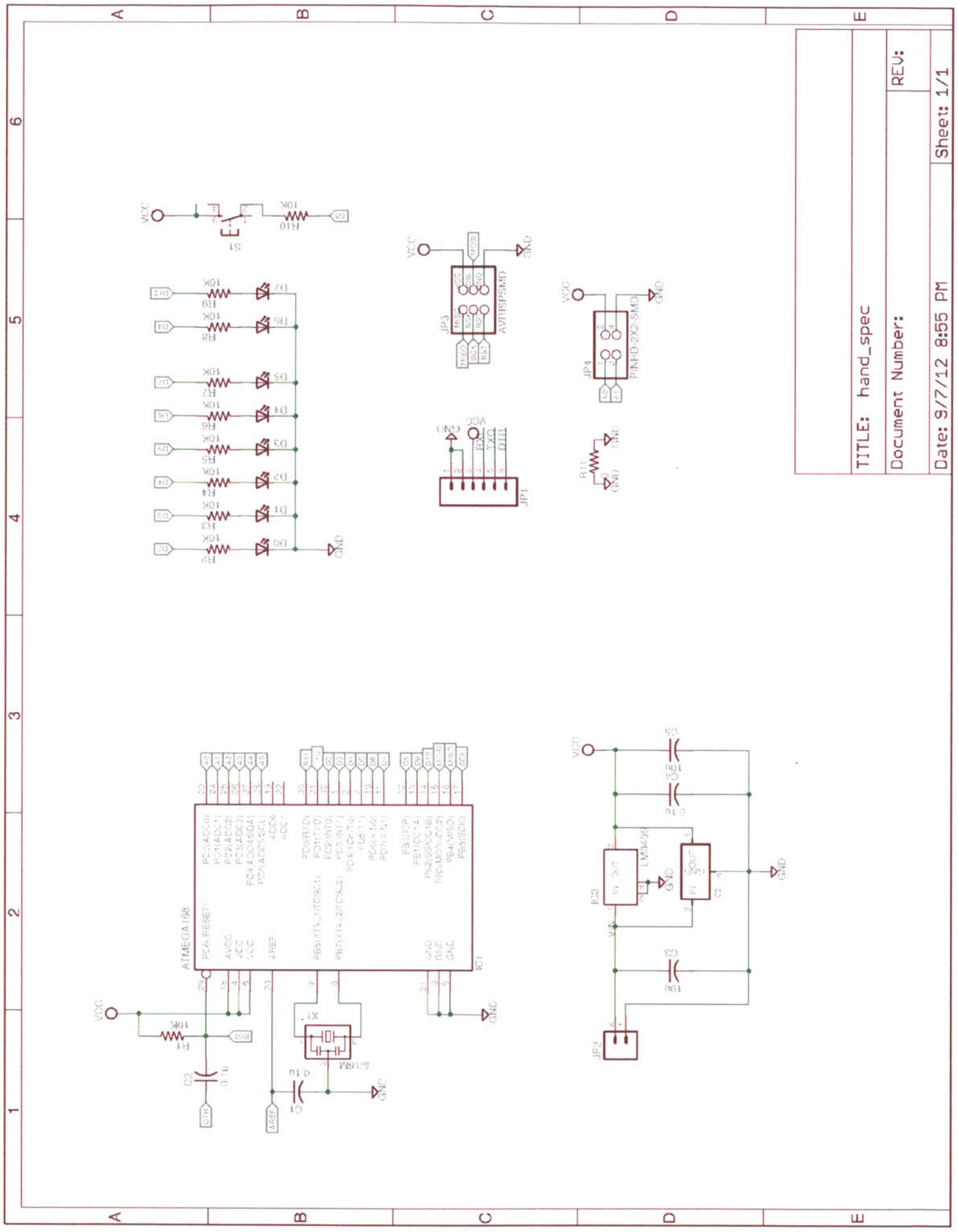


Figure G-7: Illuminator circuit board schematic.

Bibliography

[1] *Electro-Optical Kluges and Hacks A Lab Rat's Guide to Good Measurements*, October 2006.

[2] A Alemayehu. *Review of Ellipsometry for optical properties of crystals*. PhD thesis, etd.aau.edu.et, 2010.

[3] Elizabeth Allen and Sophie Triantaphillidou. *The Manual of Photography, Tenth Edition*. Focal Press, 10 edition, December 2010.

[4] J Anderson. Method to assess plant stress using two narrow red spectral bands. 2003.

[5] Arduino. Arduino .

[6] N Bazaev and S Selishchev. Noninvasive methods for blood glucose measurement. *Biomedical engineering*, 2007.

[7] Burkhard Beckhoff, Birgit Kanngießer, Norbert langhoff, Reiner wedell, and Helmut wolff. *Handbook of Practical X-Ray Fluorescence Analysis*. Springer, 1 edition, July 2006.

[8] DJ Brady. *Optical Imaging and Spectroscopy*. 2009.

[9] BA DeGraff, M Hennip, JM Jones, C Salter, and SA Schaertel. An inexpensive laser Raman spectrometer based on CCD detection. *The Chemical Educator*, 7(1):15–18, 2002.

[10] digikey. Electronic Components Distributor | DigiKey Corp. | US Home Page.

[11] I Dópido, A Villa, and A Plaza. A comparative assessment of several processing chains for hyperspectral image classification: What features to use? *Proceedings of the IEEE/GRSS Workshop*, 2011.

[12] Q Du. Unsupervised real-time constrained linear discriminant analysis to hyperspectral image classification. *Pattern Recognition*, 40(5):1510–1519, 2007.

[13] L N Foner. Artificial synesthesia via sonification: a wearable augmented sensory system. *Wearable Computers, 1997. Digest of Papers., First International Symposium on*, pages 156–157, 1997.

[14] P Franken, A Hill, and C Peters. Generation of optical harmonics. *Physical Review Letters*, 1961.

[15] Limor Fried. USBtinyISP - Inexpensive USB AVR Programmer.

[16] Ben Fry. *Visualizing Data: Exploring and Explaining Data with the Processing Environment.* O'Reilly Media, 1 edition, January 2008.

[17] N Gershenfeld. MIT CBA MAS.863 Electronic Production.

[18] Jerald Graeme. *Photodiode Amplifiers: OP AMP Solutions.* McGraw-Hill Professional, 1 edition, December 1995.

[19] Eugene Hecht. Optics, 2002.

[20] Gilbert Held. *Introduction to Light Emitting Diode Technology and Applications.* Auerbach Publications, 1 edition, December 2008.

[21] R Hellwarth. Analysis of stimulated Raman scattering of a giant laser pulse. *Applied Optics,* 1963.

[22] Philip C. D. Hobbs. *Building Electro-Optical Systems: Making It all Work (Wiley Series in Pure and Applied Optics).* Wiley, 2 edition, August 2009.

[23] Hobbs, P.C.D. Photodiode front ends: The REAL story. *Optics and Photonics News,* 12(4):44–47, 2001.

[24] Francis Jenkins and Harvey White. *Fundamentals of Optics.* McGraw-Hill Science / Engineering / Math, 4 edition, December 2001.

[25] Lite-ON. Lite-On LTW-150TK White LED, May 2010.

[26] D J Lonsdale, D A Andrews, and T A King. Single mode operation and extended scanning of anti-reflection coated visible laser diodes in a Littrow cavity. *Measurement Science and Technology,* 15(5):933–938, April 2004.

[27] K H Lundberg. The history of analog computing: introduction to the special section. *IEEE Control Systems Magazine,* 25(3):22–25, June 2005.

[28] D Mellis. FabISP - Fab-able In-System Programmer.

[29] DAB Miller. Huygens's wave propagation principle corrected. *Optics Letters,* 16(18), 1991.

[30] National Instruments. *NI-VISA User Manual.*

[31] National Semiconductor. LM3404 1.0A Constant Current Buck Regulator for Driving High Power LEDs. National Semiconductor.

[32] I Newton. *Opticks: or, a Treatise of the Reflexions, Refractions, Inflexions and Colours of Light also Two Treatises of the Species and Magnitute of Curvilinear Figures.* October 1676.

[33] ObjectiveDevelopment. V-USB - A Firmware-Only USB Driver for Atmel AVR Microcontrollers.

[34] Ocean Optics. *USB4000 Fiber Optic Spectrometer.*

[35] E Purcell and H Torrey. Resonance absorption by nuclear magnetic moments in a solid. *Physical Review,* 1946.

[36] Leonard Richardson. Beautiful Soup.

[37] Paul Scherz. *Practical Electronics for Inventors.* McGraw-Hill/TAB Electronics, 1 edition, April 2000.

[38] Rohde Schwartz. *Development Hints and Best Practices for Using Instrument Drivers.*

[39] VIshay Semiconductors. TEMD5080X01 Silicon PIN Photodiode. Vishay Semiconductor.

[40] Seoul Semiconductor Corporation. Seoul Semiconductor Corporation P7 Ultrabright White LED, April 2008.

[41] David O Sparkman. *Mass Spectrometry Desk Reference.* Global View Publishing, 2 updated edition, June 2006.

[42] R Stolen. Parametric amplification and frequency conversion in optical fibers. *Quantum Electronics,* 1982.

[43] D Streefland. USBtiny.

[44] Agilent Technologies. Agilent 34401A 6.5 Digit Multimeter User's Guide.

[45] Agilent Technologies. *Agilent E3631 DC Power Supply.*

[46] Texas Instruments, Incorporated. 2.7-V To 5.5-V Low-Power Dual 12-bit Digital-to-Analog Converter with Internal Reference and Power Down. Texas Instruments.

[47] Texas Instruments, Incorporated. Application Note 1515 A Comprehensive Study of the Howland Current Pump. Technical report.

[48] MJ Weber. *Handbook of lasers.* 2001.

[49] DG Whitehead, I Mitchell, and PV Mellor. A low-resolution vision sensor. *Journal of Physics E: Scientific Instruments,* 17:653–656, 1984.