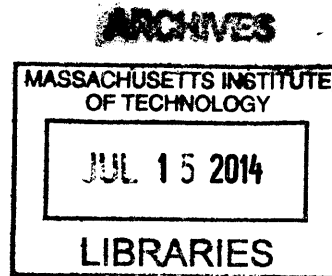


Optically Encoded Physical Keys

by

Kyle Hounsell

S.B., Electrical Science and Engineering
Massachusetts Institute of Technology, 2013



SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

MAY 2014

[June 2014]

Copyright 2014 Kyle Hounsell. All rights reserved.

The author hereby grants to M.I.T. permission to
reproduce and to distribute publicly paper and
electronic copies of this thesis document in whole and
in part in any medium now known or hereafter created.

Author:

Signature redacted

Department of Electrical Engineering and Computer Science
May 23, 2014

Certified by:

Signature redacted

Dr. James W. Bales
Assistant Director and Instructor, MIT Edgerton Center
Thesis Supervisor

Signature redacted

Accepted by:

Prof. Albert R. Meyer
Chairman, Masters of Engineering Thesis Committee

Optically Encoded Physical Keys

by
Kyle Hounsell

Submitted to the Department of Electrical Engineering and Computer Science

on May 23, 2014 In Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

Lock based security has been used since the early days of human history. Whenever people have wanted to keep their possessions safe, they have used locks to hinder those who would want to access their belongings against their will. As such, an arms race has ensued, consisting of ever more complex locks, and lock-defeating methods. This thesis is not the first time that optics have been used in locking mechanisms, but it puts forth a robust, versatile, and economical security system employing optics based on the spirit of physical keys. The proposed system uses a physical key with embedded optical fibers routed between optical ports on its surface. The corresponding reader scans the key by sequentially illuminating each port, and observing where the light exits the key. The reader then builds a matrix representing the internal connections of the key, and compares it to each key's unique identifying matrix to determine whether to grant or deny access to the current user.

Thesis Supervisor: Dr. James W. Bales

Title: Assistant Director and Instructor, MIT Edgerton Center

Table of Contents:

- 1 Introduction 4**
- 1.1 The Need for Security4
- 1.2 The Concept behind this Design5
- 1.3 Previous Work6
- 1.4 New Features and Objectives.....6
- 1.5 Future Objectives7
- 2 Operating Principles 8**
- 2.1 Sensor / Emitter Duality8
- 2.2 The Photovoltaic Effect9
- 2.3 Optical Fibers10
- 2.4 Conceptual Overview of Code13
- 3 Device Design 15**
- 3.1 Optical Design15
- 3.2 Mechanical Design and Assembly18
- 3.3 Electrical Design and Assembly21
- 3.4 Code Design and Testing23
- 4 Results 30**
- 5 Future Improvements 31**
- 5.1 Future Mechanical Improvements31
- 5.2 Future Optical Improvements32
- 6 Summary 36**
- 7 Contributions 37**
- 8 Appendix 38**
- 9 References 56**

1.0 Introduction

1.1 The Need for Security:

Security of physical systems is a longstanding concern. With the oldest example of a lock dating to almost 2,000 BC, people have been trying to keep their belongings out of the hands of others for a substantial amount of time; over the years, the technology for this has evolved substantially. Unfortunately, the art of lock picking evolved shortly after the art of lock smithing, and the result has been a continuous arms race between lock makers, and lock breakers for technological superiority in the field. Innovation after innovation have been put forth to try to combat theft, including magnetic stripes, radio frequency identification (RFID), fingerprint scanners, retina scanners, and others, but each has demonstrated an inherent flaw of some sort or another. Locks can be picked, and indeed the infrastructure for doing so is in part included in legitimate lock smithing.

Recent work by UC San Diego researchers¹ has demonstrated that standard keys can be duplicated from a mere photo of the same sitting on a desk, taken from a football field away by any number of readily available commercial cameras with telephoto lenses, and a piece of software called Sneakey. Fingerprints can be easily stolen, and RFID cards can be sniffed at a distance with a backpack full of hardware². Malicious motives aside, some failures in security systems result from the physical frailty of a system. Mag-stripes are prone to failure by physical wear; you only have to look so far as your credit card to see that. Another common failure is snapping a key in a lock. Password locks are only as good as the memory, and discretion of the user, with problems such as old passwords, common passwords, and passwords shared across devices and services being potential pitfalls to the user. On a physical entry system, keypads are

often located in an exposed area next to or on the door, giving rise to the possibility of a malicious observer recording the keystrokes, for their own nefarious use later.

1.2 The Concept behind This Design:

The main idea of the proposed design is using a physical token with embedded optics as a key. The key can be passive or active; however, a passive token will be more secure by virtue of not giving off any detectable radiation to indicate either its presence or its operating principles. The system discussed in this paper is of the passive kind. In the simplest form, the token is a small stick of material with optical ports on one face of the token and embedded optical paths interconnecting selected pairs of ports. A key reader uses light sensors and emitters to map the internal structure of the key, generating a connection matrix for the key's ports. The generated matrix is then compared to a database of matrices for the keys of authorized users, to determine whether or not to grant the user the desired access to the protected materials/space.

Because the optical key does not rely on semiconductors or magnetic storage, it is resistant to degradation or destruction by errant magnetic fields. Additionally, the key need not be made with any metal at all, lending itself to applications inside hospitals with magnetic resonance imaging (MRI) scanners, or inside laboratories with high magnetic field experiments, including Bitter magnets and Tokamaks. To copy the key, one would need to obtain the key itself, and build a scanner capable of reading it, as simply photographing it would not reveal its internal structure.

1.3 Previous Work:

The first work I could find that is similar to the concept I propose in this paper, is US patent #4090175³, issued in May of 1978. The system it describes is a direct optical analog to punch cards. Each key was a metal sheet with holes drilled into it, that either allowed or blocked light, shining on one side of the card, to reach optical sensors on the other side. As the holes were drilled in a grid, and simply went through the card in a predictable fashion, the key would be quite easy to duplicate from a photograph, or even a sketch. Additionally, since the keys were designed to be small sections of thin metal sheet, they would not be especially resilient. This problem would be further exacerbated by the holes drilled through them, which would make them easier to bend or snap.

The second work that I could find dealing with this concept is detailed in European patent EP 1 272 721 B1⁴, issued to Jeremy Rice in August, 2003. It describes a similar system to what is outlined in this paper, however the routing of the fibers inside the key and lock bodies is done in a very haphazard fashion. Additionally, the key scanning procedure is implemented in a different, less efficient way, which does not allow for several features described in this paper, which provide extra security to the system.

1.4 New Features and Objectives:

The design outlined in this paper takes several steps away from the previous designs in improving usability, manufacturability, economy, and security of the lock system. In short, the system is designed to incorporate more common materials in its manufacture, as well as require a more standard set of skills from those working on the hardware and software design end. By not necessitating the use of a specialized programming language, or non-standard chips, workers on

the development side of the device would not require time consuming training that was only good for the development of this device, instead either coming into the job already having, or leaving it with a skill set that is useful in other industry applications. Additionally, several included features make this optical key design use less material to greater effect than previous designs have been able to achieve, reducing production costs without compromising the security of the system.

1.5 Future Objectives:

Development of the system beyond this work will primarily focus on adding additional security to the system, and sorting out the requirements for marketability. Security wise, the system has much room for growth by employing various optical tricks, including components such as dichroic materials, and polarizers to selectively re-direct light based on qualia such as polarization and wavelength. Concerning marketability, manufacturing techniques such as 3D printing could drastically cut the cost of the key systems manufacture and upkeep.

2.0 Operating Principles

2.1 Sensor/Emitter Duality:

The proposed design exploits the dual nature of the LEDs, which can act as both photo-emitters and photo-detectors. So, instead of having a separate emitter and detector for each conduit in the key, as in previous designs, each optical port on the key is coupled to an LED when the key is inserted into the reader. Each LED acts in both the emitter and detector roles. To detect the interconnections between the ports, the reader drives the LED on the first port as an emitter and uses the others as detectors to see which ports connect to that first port. Then the system drives the LED of the second port as an emitter, using all others to detect which ports are optically connected to port two, and repeats the process for all of the remaining optical ports.

This dual-use of the LEDs means that the optical pathways of the key can now be probed in both directions, which allows for more combinations per optical port in the key, and thereby increases the overall security of the system. (The older systems mandated that each port had to be an emitter or detector, therefore connections had to be between an emitter port and a detector port, and never between two emitters or two detectors.) Additionally, the use of a single low cost module for both the emitters and detectors, combined with the lower number of optical ports required to achieve the same number of codes, means this design can be manufactured with fewer parts and lower component costs.

While it is technically possible to achieve a similar bidirectional capability using discreet emitters and detectors in combination for each optical port, an optical beam-splitter would be required for each pair. This would increase both cost and complexity, as well as reduce signal strength by 75%, with no clear performance gain.

2.2 The Photovoltaic Effect:

The dual use of the LEDs is a consequence of the photovoltaic effect. This effect causes a semiconductor junction that is exposed to light to produce a voltage across itself, and thereby be able to source a current when placed in a circuit. The root of this effect is the mechanism by which the junction absorbs a photon with an energy greater than or equal to that of the bandgap and forms an electron/hole pair, by exciting an electron from the valence band, into the conduction band.

To efficiently detect incident light, the LED must have a high photovoltaic efficiency. Here the effect is exploited along with the fact that a reverse biased junction has a measurable capacitance that is directly proportional to the area of the junction. To sense light, an LED is first reverse biased to build up a voltage across its reverse bias capacitance. Then, while illuminated, the photo-current discharges the junction capacitance. Since the photo-current is proportional to the irradiance of the junction, a brighter irradiance causes the capacitor to discharge more quickly. Thus, the more light an LED has received since the source of the reverse bias has been removed, the lower the voltage across it will be when measured after a set delay. Given that the junction has not been completely discharged, an LED exposed to light will have a lower voltage than one kept in the dark for the same amount of time.

Any light leaking into the reader and striking the LED will also discharge the capacitor. By choosing an LED with a large die, the capacitance of the junction goes up, allowing the reverse biasing voltage to remain higher longer for a given light leak, reducing the effect of the light leak. The photo-current also increases proportionately with die area, so the sensing time is independent of the die area, so long as the light is of uniform intensity.

2.3 Optical Fibers:

When evaluating optical fibers for their use in the project, two properties of optical fibers were most significant in my research; the numerical aperture and minimum bend radius of the fibers. The numerical aperture of a fiber refers to the cone of light that the fiber will accept for propagation. It is pertinent to the design because the acceptance angle of the optical fibers places constraints on what LED can be used. If the LED gives off light in a cone with an angle greater than the angle of the cone of light that the fiber will accept, the coupling between the two will be leaky and inefficient. The minimum bend radius of a fiber refers to how narrow a circle a fiber can be bent into before breaking. This radius directly controls the size of the key. To contain the fibers, the key must be physically larger than the minimum bend radius of the fiber, so that the fiber can make a 90° turn coming from one optical port, run down the length of the key to the next optical port, and make another 90° bend to exit the key. Furthermore, it limits the spacing of adjacent optical ports on the key, as a fiber must complete 180° of curvature to go from one hole to another, though circumventing this could conceivably be accomplished by routing the fiber away from or past the adjacent optical port, completing a 180° turn elsewhere in the key, and returning to the adjacent port, though this strategy requires the key to be able to accommodate double the minimum radius of curvature. If arranged so that the 180° bend were placed on a diagonal of the cross section, the width of the key would have to be increased to at least $\sqrt{2}$ times the minimum bend radius of the fiber.

Light transmission in an optical fiber is based on a phenomenon known as “total internal reflection.” Total internal reflection arises from the difference in indices of refraction between two media, in this case, the core and cladding layers of a multi-mode step indexed optical fiber.

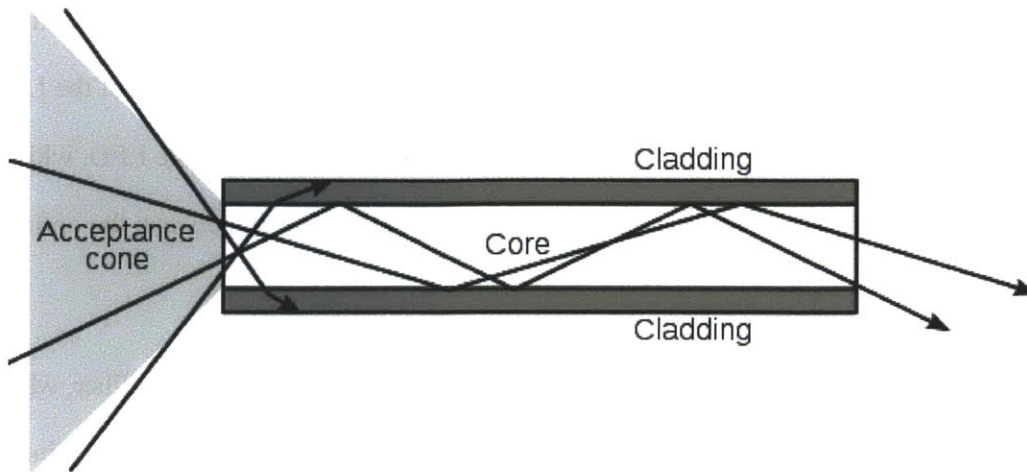


Figure 1, The acceptance cone of optical fibers is governed by the difference in refractive indices of the core and cladding. Light entering at an angle greater than that of the acceptance cone is not totally internally reflected. Figure retrieved from Wikipedia⁵.

A light ray approaching the fiber from the surrounding medium with an index of refraction n is only accepted by the fiber if it falls within the acceptance cone of the fiber as illustrated in Figure 1. The maximum angle from the fiber axis that can be accepted is given by

$$n \sin \theta_{\max} = \sqrt{n_{\text{core}}^2 - n_{\text{clad}}^2}$$

The quantity on the left side of the equation is known as the “numerical aperture” of a fiber, and as stated by Anchor Optical⁶, for a typical acrylic fiber it is equal to approximately 0.5. As the light rays in the key reader will be incident on the fiber from air, which has an n of 1.00, the numerical aperture is equal to the sine of the maximum angle of light incidence that the fiber will accept, in this case 30° , the half angle of the maximum acceptance cone of 60° . For a glass fiber, the numerical aperture spans a range from .23 to .66 depending on the composition^{7, 8}, which corresponds to an angular acceptance from 13° to 41° . Since LEDs are manufactured with emission cones as small as 5° , the numerical aperture of the fiber is not a definitive test for compatibility with the system, as all of the fibers mentioned earlier are capable of accepting all of the light coming out of an LED with up to a 13° emission cone half angle. That being said,

acceptance angles greater than 20° do allow more flexibility when choosing an LED because of the wider variety available. If the acceptance angle of the fiber is greater than that of the LED, it could also cause the light exiting the fiber to project onto an area wider than the LED, which would reduce intensity somewhat, but also allow for some misalignment between the optical fiber and the LED.

The second property of interest is the minimum bend radius of the optical fiber, which is a measure of how sharply a fiber can be bent before it sustains a fracture. I was able to find numbers for bending Corning 125 μ m diameter glass optical fiber on their website, which listed 5mm as their minimum recommended bend radius⁹. Their measure of “minimum” however, is the bend radius that the fibers can conform to without incurring a fracture in N loops, and for a 5mm minimum bend radius, N was equal to 10^5 . While this is a good metric for commercial shipping and installation, it is a bit conservative for use in the consideration of fibers for this project, later discussed in section 3.1. Numbers of this type were harder to come by for acrylic fibers, so later on in the project I determined them experimentally.

Additionally, as the diameter of the optical fiber increases, so does its minimum bend radius. In order to gather more light than a single small fiber, but have a smaller minimum bend radius than a single large fiber, I chose to employ fiber bundles, which package 50+ small fibers together in a plastic sheath. The use of small fibers allows them to slide past each other when bent, relieving tension that would have caused a fracture in a single fiber of equivalent diameter, and the constituent fibers can bend to a much smaller radius than a single fiber. Bundling also allows for the transport of much more light than a single fiber would have.

2.4 Conceptual Overview of Code:

Building off of sections 2.1 and 2.2, the code for this device exploits the photovoltaic effect and the duality of LEDs as both sensor and emitter at its basic level. To determine the internal structure of the key, the reader needs to scan and map the key using its LEDs. To do this, the key is inserted into the lock, which is reasonably light tight, and the scan is initiated. To scan, the device configures its LED connected pins as outputs, reverse biasing all but one of the LEDs, causing them to build up a charge at their semiconductor boundaries (depletion layers). The last LED is forward biased, causing it to turn on, and illuminate the optical port of the key that is adjacent to its port. Then, the microcontroller changes all of the high side pins of the reverse biased LEDs to high impedance inputs, and waits. While the microcontroller waits, the light from the forward biased LED illuminates one or more of the reverse biased LEDs. This illumination generates a photocurrent that discharges the LEDs. If the microcontroller waits a sufficient amount of time, the LED voltage on the illuminated LEDs will fall below the digital high threshold, while the non-illuminated LEDs will register above it. The results of the read are then stored in memory for use later in the code.

If the microcontroller does not wait long enough before reading the reverse biased LEDs, the voltage across the illuminated LEDs will still register above the digital “high” threshold for the microcontroller. If the microcontroller waits too long, all of the LEDs will register below the threshold, as the thermally-induced junction current in the non-illuminated LEDs will discharge them over a much longer time scale than the photocurrent.

To fully scan the key, the process is repeated, each time changing which LED is forward biased until all LEDs have been forward biased. The results of all of the scans are arranged into a

matrix, which forms a map of the optical interconnections in the key. This matrix is compared against the matrices of keys of authorized users, and if it matches one of them, the lock opens.

3.0 Device Design

3.1 Optical Design:

As mentioned in section 2.3, the minimum bend radius of optical fibers is of critical importance to the system. Corning provided minimum bend radius numbers for single glass optical fibers that kept the failure rate below 1 in 100,000 loops. While this is a good measure for a commercial standpoint, where only one fiber is being run from a junction box to a home, or between two servers in separate buildings it is less applicable in this device, which uses jacketed bundles of fibers; each bundle is a group of tens of fibers held together by a plastic jacket. The fibers in each key will only complete a total of $H \cdot F / 4$ full loops where H is the number of ports on the key, assuming only one connection per pair of optical ports, and F is the number of optical fibers in each connecting bundle. For a key with 8 optical ports and bundles of 60 fibers, this is 120 loops, which, with Corning's numbers applied yields a fracture rate of one failure per 416 keys. As the system is designed to have 59 fold redundancy for each fiber by virtue of bundling, this number, which initially might sound like a high failure rate, is beyond simply tolerable, as a failure rate of 1 in 3 loops would translate to a failure rate of 1 in 6 fibers; easily absorbed when there are 50 other fibers in each bundle capable of carrying light. Given this information, it is easy to imagine that a much smaller radius of curvature could be used in the key, which is important, as the minimum bending radius of the fiber dictates how large the key must be in order to contain the loops of fiber without breaking them.

It was more difficult to find data for acrylic optical fibers I ordered a sampler pack from Anchor Optical in order to divine their characteristics myself. I split open one of the larger jacketed fiber bundles of the type that I was most likely to use for the project. After extracting

the fibers, I measured the diameters of ten of them so that I could average the readings into a more representative number. The average was easy to calculate, because with digital calipers precise to 0.01mm, all ten fibers measured 0.25mm in diameter. Next, I wrapped a few of the fibers five times each around successively smaller mandrels to ascertain when fractures were incurred, where fractures were visible breaks in the fiber, complete or partial, under a 15x loupe. When no visible fractures were incurred at a .375mm bend radius for any fibers in the sample, I stopped testing. This data makes acrylic optical fibers an enticing alternative to glass ones, though I do not have a failure metric as refined as Corning provided. To test, I bent a jacketed fiber bundle, which created a terrifying crackling sound as the outside and inside fiber strands slid past each other; however my subsequent inspection of the fibers revealed no actual failures. It is possible that smaller scale fractures were incurred during the bending, but I was unable to observe either gross attenuation of light passed through the fibers, or glowing patches along the length of the fibers that would indicate breaks in the fiber tunneling light out.

It is useful to note that, in the case of severe bends, the pressure change on the outside and inside materials of the fiber is enough to alter the fiber's index of refraction. This, combined with the fact that the extreme curve of the fiber does not perfectly mesh with a straight cylinder propagation approximation of an optical fiber, results in the loss of higher propagation modes of the fiber. This change corresponds to a narrowing of the effective acceptance cone of the fiber and a reduction of the effective numerical aperture.

For the prototype, the optical arrangement I created was designed to be simple and hassle free. To assure good coupling of light from one LED to the others, I decided to use 3.2mm diameter acrylic optical fiber bundles, the largest diameter bundles that I had ordered in the sampler kit. As discussed previously, multi-mode graded index acrylic fibers were chosen for

this application, due to both their high tolerance for mechanical abuse, and their wide acceptance angle. Their ability to tolerate an extremely small bend radius allowed me to position the fibers such that their final 4mm of length were held precisely perpendicular to the face of the key, ensuring the alignment of the acceptance cone of the fibers with the emission cones of the LEDs. Metal was chosen for the housing of this system because its reflectivity is advantageous for the LED-to-fiber part of the optical path, in that it re-directs some of the light from the LEDs that would otherwise have been lost to absorption and scattering upon incidence with the walls of the coupling chamber, back into the optical fiber. This again exploits the wide acceptance angle offered by multi-mode acrylic optical fibers. It follows that in this case, using LEDs with an emission cone half angle of 45° or less, all light incident onto the surface of the optical fibers either directly emitted from the LEDs, or reflected off of the coupling chamber walls as a primary reflection from the LEDs will be successfully coupled into the optical fiber bundle's fibers. A metal housing is additionally beneficial due to its opacity. The use of high opacity housing materials is essential because it prevents any cross talk between optical ports. If the housing were transparent, light emitted from one LED could potentially spill into the optical fibers of adjacent optical ports, which would yield incorrect results in a scan. This type of light leak could also directly expose adjacent LEDs that are in the detection state, producing a false positive for detected light on those affected.

3.2 Mechanical Design and Assembly

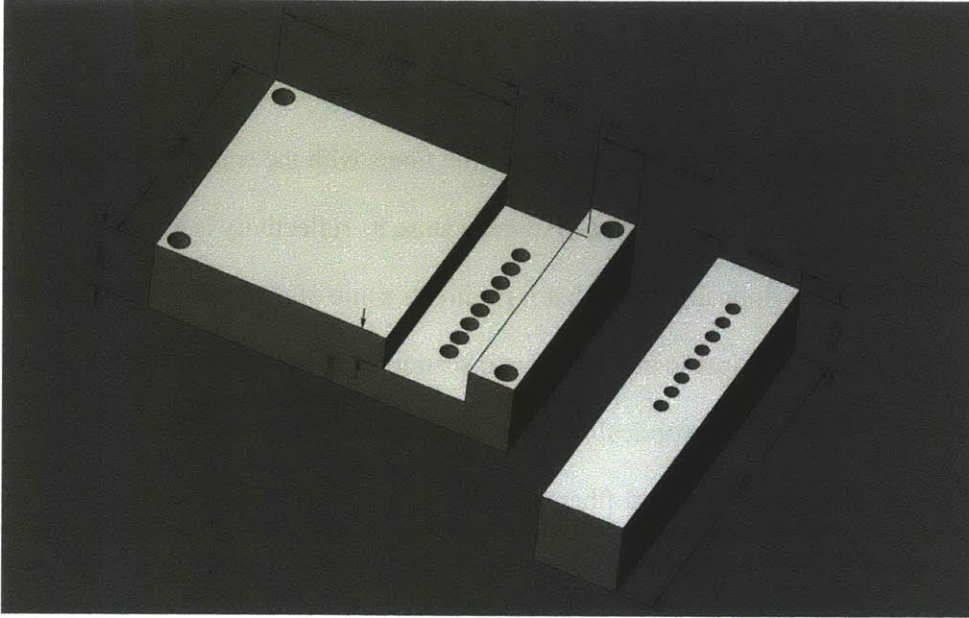


Figure 2, Mechanical model of large scale prototype. Dimensions in mm.

For simplicity's sake, I started off the mechanical assembly with minimal extra features, and a large scale. The first prototype version of the circuit uses 8 LEDs for its scanning optics, and all monochromatic red LEDs in standard 5mm packages. The system uses the simplest design for the optical ports for the key, arranged linearly down the center of the keyway, as shown in Figures 2 and 3. The key reader assembly was fabricated from 1"x3"x5" aluminum stock, by milling a 1" wide keyway into the stock across its 3" dimension and drilling 8 holes into it at 0.1" intervals on one side of the keyway (Figure 3).

The holes were sized to accommodate the bulk of the LEDs, but not admit the thin skirt around the base of the LEDs. This ensured that the LEDs were all oriented correctly and a uniform distance from the front surface of the key. The 1" size of the keyway was chosen to allow ample room inside the first key prototype for fiber routing and other operations. Mating holes were drilled into the key after transferring the original holes with a center punch

beforehand. The connections for the prototype key were made between holes 0&7, 1&4, 2&6, and 3&5, with hole 0 being the hole closest where the user holds the key, in order to get a variety of different radii of curvature to test the optical fibers with. Figures 4 and 5, show the key with hole 7 on the left, and hole 0 on the right.

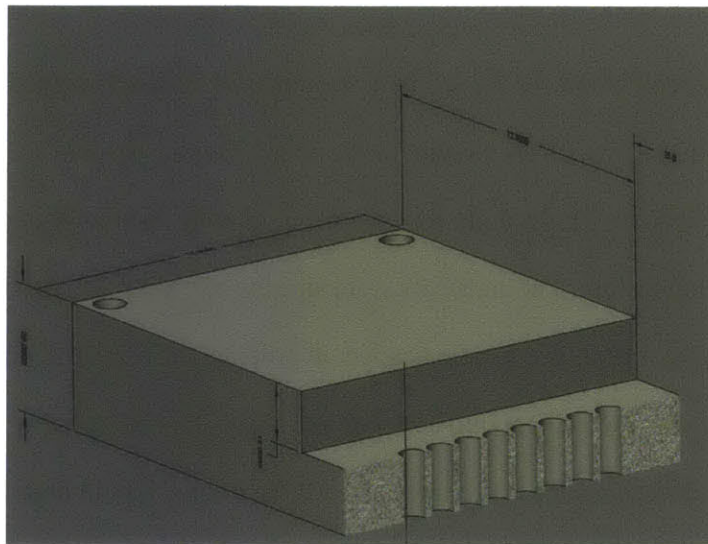


Figure 3, Cutaway view of LED sockets of reader.



Figure 4, Cutaway view of large scale prototype key.

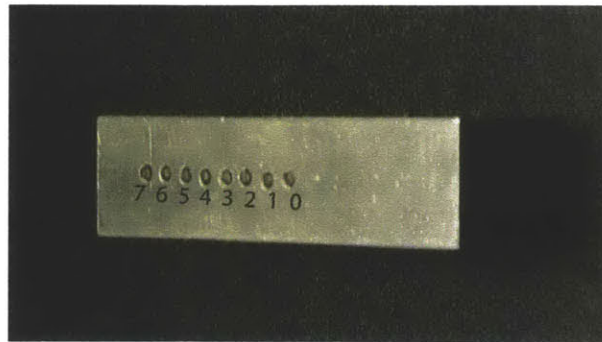


Figure 5, Prototype key in 1" aluminum.

The printed circuit board for the reader is mounted on standoffs tapped into the aluminum block. Instead of soldering the LEDs directly to the board, Female headers are used to attach the LEDs, as shown in Figure 6. This feature allowed me to quickly and easily exchange LEDs of different flavors during testing. The standoffs were sized to allow a 0.25" gap between the metal mounting block and the tops of the female headers in order to allow for LED mounting flexibility. The LED wires were bent into a spring like shape in order to apply pressure and keep the LEDs snug in their receptacles at all times, even if the printed circuit board was jostled.

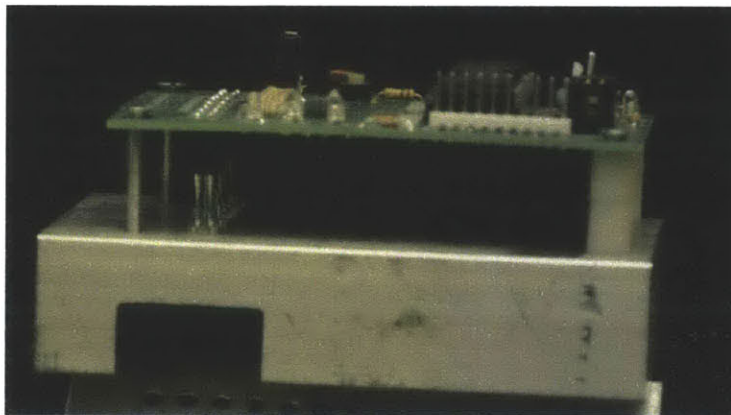


Figure 6, Circuit board installed on lock housing, with LEDs mounted on standoffs.

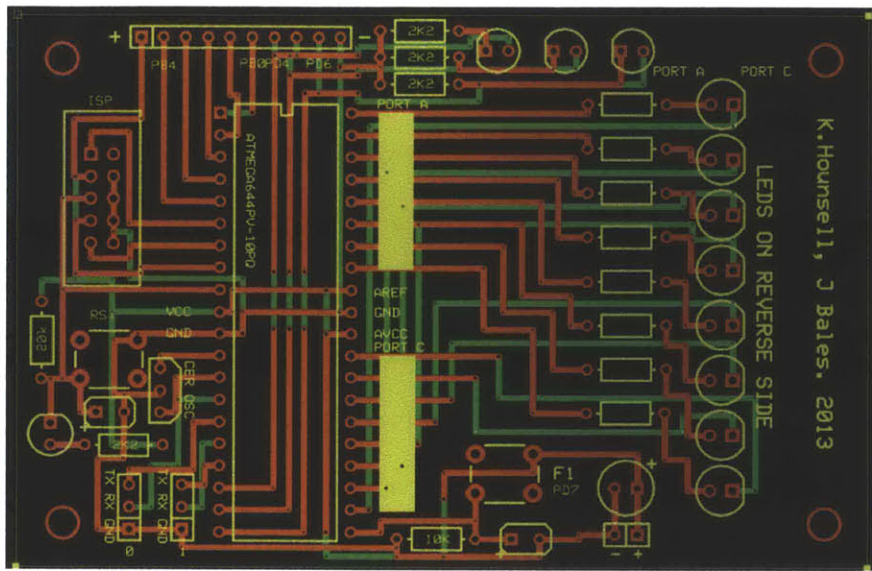


Figure 8, Layout of circuit board in "Express PCB" editor. Green corresponds to bottom copper, red to top copper, and yellow to silk screen.

In choosing a matching resistor several factors needed to be considered, including the current rating of the LEDs, and their reverse junction capacitance. The LEDs are Vishay semiconductor type TLDR5800. Their forward current rating is 20mA, with a typical luminous intensity of 2500mcd at 648nm wavelength. With a forward voltage of 1.8V, and an output drive voltage from the ATmega of 5V, the current limiting resistor needed to be greater than 160Ω, so I chose 220Ω. The diodes tolerated 6V reverse voltage, which is good, because in order for the device to function, they need to be reverse biased to 5V. As discussed in the optics section, this model of LED was chosen for its narrow emission cone, having a 4° half angle. Finally, these LEDs were also desirable for their medium junction capacitance, typically 50pF, which allows them to discharge their reverse biased capacitance reasonably quickly under illumination. Additionally, this capacitance and the current limiting resistor combined set the time needed to charge the junctions to their reverse voltage. Using a time that is five times the product of the resistor and capacitor's values allows the capacitor to charge to approximately 99% of its final value. Here the time constant comes out to 55ms, which equates to a 441ms lower bound on the read time.

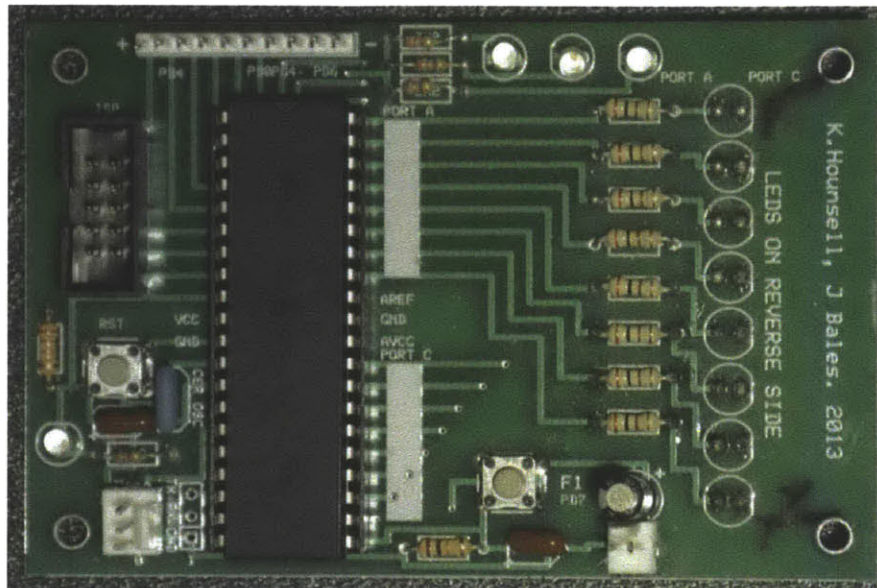


Figure 9, Populated circuit board.

3.4 Code Design and Testing

Conceptually, the software running this project is not particularly difficult. Exploiting the photovoltaic effect simply required the appropriate port manipulation. To scan, the device configures its LED connected pins as outputs, reverse biasing seven of the eight LEDs, and forwards biasing the last one. This bias causes one of the LEDs to turn on, and shine into its fiber optic port, and the other seven to remain off, but build up charge at their semiconductor boundaries. Once the LEDs' reverse capacitances are charged, their control pins are switched to high impedance inputs, so that the microcontroller does not discharge the junctions on its own. The microcontroller then waits for a set period of time, and during this time, the charge on the junction capacitances of the reverse biased LEDs decays. The decay is asymmetrical, however, between the LEDs exposed to light, and LEDs not exposed to light. As the photovoltaic effect dictates that a semiconductor junction exposed to light with energy equal to or greater than that

of the bandgap will attempt to create a voltage differential across itself by inducing the flow of electron hole pairs, the illuminated diode junction's charge will decay markedly faster than the non-illuminated junctions' charges. Since $q=CV$ applies to an LED's junction capacitance, just like any other capacitor, as the charge on the junction is depleted, the junction's voltage drops proportionally. If the microprocessor then polls the input pins at the right time, the pin connected to the illuminated LED will register as a digital "low" or a 0, whereas the pins connected to non-illuminated LEDs will still register as digital "high," or 1 signals.

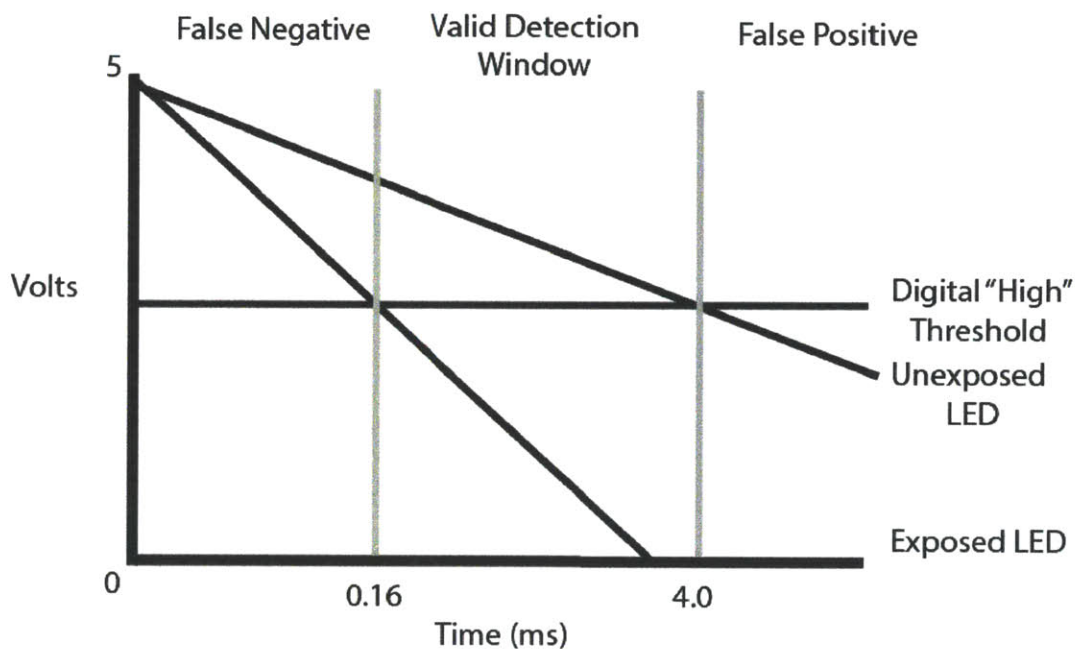


Figure 10, LED voltage vs. time for exposed and unexposed LEDs.

This strategy is illustrated in the above figure, with the leftmost sampling area corresponding to a failed read, as both illuminated and non-illuminated diodes register above the microcontroller's "high" threshold voltage, and with the center sampling area corresponding to a proper read. The rightmost sampling area also yields incorrect results, as both the non-

illuminated and illuminated LEDs register a digital “low” voltage. By simply writing the input state of the whole port to a byte of memory (for an 8 LED system), and then incrementing the illuminated LED selection, the scanner can quickly generate a representative square matrix of the internal structure of the key being scanned.

Code Iterations.

In initial testing the code (iteration 1) only used one LED as a detector. The LED was connected, along with a series resistor, between corresponding pins on two ports of the microcontroller. Both ports were initially configured as outputs, and appropriate voltages are written to the pins in order to place the LED into the reverse biased state. Next, the microcontroller pin connected to the cathode was switched over to input mode. It was important that any pull up or pull down resistors be disabled for the read cycle, or else the current they supply will either oppose the photo-current and lead to a false negative for light, or artificially deplete the charge from the junction capacitance leading to a false positive for light. Once the appropriate microcontroller pin had been switched over to the input state, the microcontroller had to wait for any present light to induce a photo-current in the device, lowering the voltage of the cathode pin as it did so. Waiting too little would not have allowed enough photo-current to flow, and again lead to a false negative for light exposure. Waiting too much would have led to voltage decay of even a covered diode, via thermal electrons and leakage current from both the diode and microcontroller input, and lead to a false positive for light exposure from the device.

The next iteration of the code involved using a second LED as an emitter, and routing its light through a piece of optical fiber to the first, which was configured as a detector. It confirmed that the fiber optic bundles that I had chosen for the project were indeed a viable way of

transporting light within the key. The success of this code led to a third iteration, which used a third LED to monitor ambient light and thereby filter out false positives.

Conflict with the JTAG Port.

At this stage conflict with the JTAG port, an embedded feature of the ATmega hampered the development of the project. Initial testing with LEDs 0,1, and 2 showed that the concept of having LEDs function as both emitters and detectors in the system was feasible. The test code forwards biased diode 0, so that it emitted light and reverse biased the other two so that they detected light. A positive reading was given if and only if diode 1 indicated that it had been exposed to light, and diode 2 did not. This filtered out false positives from a well-lighted environment, which would have caused both diodes to indicate light exposure. This test code functioned correctly when uploaded to the device, and indicated that optical fiber was a viable material for forming the light channels of the key.

I then attempted to expand the sensing capability to the rest of the port, but ran into problems. At this stage, there were problems with the middle nibble of port. The LEDs attached to it refused to light at more than a fraction of their full brightness. To debug, I created a small piece of code to sequentially activate all 8 LEDs in emitter mode, in an effect reminiscent of the Cylon eye animation in Battlestar Galactica. The behavior seemed indicative of a high impedance connection, and the fact that the problem was surfacing in the middle nibble rather than the high or low nibble suggested that the problem was arising from a feature integrated into the ATmega, rather than the code. The root cause of this problem turned out to be the JTAG debugging port on the ATmega, and in order to disable it, the JTAG disable bit need to be written inside the setup function twice within a short amount of time, as the ATmega designers had specified the use of two sequential writes within a certain number of clock cycles of each

other be required in order to prevent accidental disabling of the JTAG port by a single erroneous write, and then buried this specification in an obscure subsection of the datasheet. Once the problem was isolated, and the solution implemented, the sequential activation code performed as expected.

Light Leaks.

Once I had reviewed the code with verified full port functionality thoroughly, and was satisfied that it was conceptually sound, I tested it, and it failed to function properly. In particular, the code rejected the key whether inserted correctly or not. I was unsure of the root of the problem, but given the debugging that I had done on the three LED code, and on the eight LED scanning code, I was reasonably confident that the port manipulation and LED driving operations were not the culprit. I created an auxiliary function entitled “regurgitate()” to graphically troubleshoot this. The function displayed the contents of the memory on the scanning LEDs at the end of the read cycle, allowing me to represent the memory contents as a table of eight bytes represented with two hex characters each. Table 1, shows this concept for an ideal read of the fabricated key, where each column represents the read state of the port when the LED number listed in the top cell of the column is driven. In the table, a “1” is used to indicate an LED that indicates that it was exposed to light, and a “0” signifies an LED that does not. The diagonal row of “X”’s indicates the driven LED in each read, which will always show up as a “1” in practice, thus for the hex representations of the bytes, an “X” is treated as a “1”. In the table, LED 7 is considered the most significant bit.

Table 1, Correct read characteristics for sample key.

LED Read	LED Driven							
	0	1	2	3	4	5	6	7
0	X	0	0	0	0	0	0	1
1	0	X	0	0	0	1	0	0
2	0	0	X	0	1	0	0	0
3	0	0	0	X	0	0	1	0
4	0	0	1	0	X	0	0	0
5	0	1	0	0	0	X	0	0
6	0	0	0	1	0	0	X	0
7	1	0	0	0	0	0	0	X
HEX	81	22	14	48	14	22	48	81

Tables 2 and 3 show the results of the testing. Table 2 shows the number and type of detection errors for scans of different lengths of time. The bytes of Table 3 are arranged such that the ones in the binary representation of the nth byte represent the LEDs that were either exposed to light during the time that the nth LED was active, or emitting light during the time that the nth LED was forward biased. The code lists are arranged chronologically from left to right in the cells of the middle column. In the table, the second row shows an example of what the hex byte list would look like for a delay that was within the range of correctly functioning values. Over the course of testing, I determined that the light coming out of the key was leaking into detectors adjacent to the target detector, though not into the detectors adjacent to the injection port. I suspect that if the key and lock faces had been machined to tighter tolerances, the light leak would have been a nonissue, however the mills that I had access to have considerable backlash. To combat the reliability issue created by the light leak, I shortened the light collection period of the detectors by a factor of 100, which decreased their sensitivity, and yielded more reliable data as shown by the decrease in errors shown in Table 2. The downside to this fix is that it hampers the ability of the device to use fibers with a smaller cross section, which hurts the capacity of the

device to be miniaturized. Additionally, this trick cannot be arbitrarily extended, as once the delay time is lower than 160µs, the reader begins to accumulate errors again.

Table 2, Error characteristics for selected read delay times.

Delay Time	False Positives	False Negatives	True Positives	True Negatives
64ms	28	0	16	20
32ms	19	0	16	29
16ms	11	0	16	37
8ms	7	0	16	41
4ms	0	0	16	48
1.6ms	0	0	16	48
800µs	0	0	16	48
400µs	0	0	16	48
160µs	0	0	16	48
80µs	0	8	8	48
40µs	0	8	8	48
8µs	0	8	8	48

Table 3, Results of key reading for different read delay times.

Delay Time	Regurgitated Code List	Works?
A Functional Value	81,22,14,48,14,22,48,81	Yes
64ms	F9,7A,7C,FA,7E,7E,78,F9	No
32ms	E1,7A,7C,78,3C,7A78,E1	No
16ms	81,72,3C,78,3C,62,78,81	No
8ms	81,72,3C,68,1C,22,68,81	No
4ms	81,22,14,48,14,22,48,81	Yes
1.6ms	81,22,14,48,14,22,48,81	Yes
800µs	81,22,14,48,14,22,48,81	Yes
400µs	81,22,14,48,14,22,48,81	Yes
160µs	81,22,14,48,14,22,48,81	Yes
80µs	01,02,04,08,10,20,40,80	No
40µs	01,02,04,08,10,20,40,80	No
8µs	01,02,04,08,10,20,40,80	No

4.0 Results:

Over the course of design and assembly, several critical concepts and characteristics of the device were validated or characterized. In designing the device, I characterized the sweet spot for choice of read delay times, in order to avoid both false positives and false negatives, given the choice of LED. Primarily, however, I was able to demonstrate that the design that I set out to build was indeed feasible to build with commonly found parts, and on a small budget. I tested the prototype key and lock system through fifty cycles with the key inserted either properly, improperly, or not at all, and with the correct read delay dialed in, there were no inaccurate results of either kind in the fifty cycles. The lock showed no signs of wear, and also tolerated some misalignment of the key in the keyway with no impact to functionality. Overall, testing seems to indicate that the design is sound.

5.0 Future Improvements

As the prototype was designed to be a proof of concept, it is a barebones system. In the future, many steps can be taken to greatly improve the security, and manufacturability of the device.

5.1 Future Mechanical Improvements

I fabricated this prototype with ease of construction in mind. It would be easy to add further security to the key system with mechanically simple features. Potentially the easiest way to reduce compatibility between locks in different installations would be to rearrange the pattern in which the holes of the key and reader are drilled. This would cause keys from one lock to be incompatible with another reader, as the fiber ports of the key would not line up with the LED ports on the reader. The simple 8 LED linear arrangement was solely done for ease of manufacturing.

In the same vein, the key cross section need not be a square. A triangular or hexagonal cross section would work equally well, and again preclude one installation's lock from accepting another's key, and vice versa. In my opinion, it is worth noting that a circular cross section without any sort of indexing would be a sub optimal solution, as it would present the user with a large potential for misalignment based errors, especially if a smaller key cross section and fiber cross section were used. Rotating the key would also be non-viable, as the key would need to be still and in alignment for the entirety of a scan cycle, which could last several seconds.

A third technique would involve routing the optical fibers between different faces of the key. This would make manufacturing slightly less practical, as the fibers would now need to be routed in more than a simple x-y plane. Routing could likely be accomplished by pre-cutting and

bending the optical fibers before insertion into the key housing, and after drilling the receptacle holes in the key body, as well as drilling out a central chamber for the fiber bundle, using a pick and place style machine to insert the fibers into the confines of the key. An epoxy based filler would function well as a securing agent for these fibers.

5.2 Future Optical Improvements

Much as the mechanical elements of the device can be modified in multitudinous ways to enhance the security of the system, the optics of the device can also be modified to the same effect. While the prototype device used solely red LEDs, future iterations of the device might employ multiple colors of light to great effect when coupled with the use of dichroic optics to separate the optical paths within the device by color. This implementation would greatly increase the potential number of codes per number of optical ports in the key, but would also increase the scanning time for each key, as a separate scan would have to be run in each color. Dimension wise, keys implementing dichroic filtering would not have to be significantly larger than non-dichroic keys; the main fibers would remain unchanged, and the dichroic filters and splitters could be implemented as small in-line modules. Additionally, the details of LEDs' bandgaps are pertinent here, as a direct-bandgap green LED would not be able to detect light from a red LED, as the energy of the photons of red light is too small to excite an electron across the bandgap of a direct green diode. This can be used to the advantage of the reader, in order to discriminate light incident to a detector based on color, e.g. if the red, green, and blue diodes register incident light during a scan, the light contains blue light. If only the red diode registers incident light during a scan, the light contains neither blue nor green light, and so it must be only red light.

Another way to increase the security of the device by using improvements to the optical system would be the inclusion of polarizers in the system, and polarization preserving fibers in

the key. If the light emitted by the locks were polarized, the keys for those locks could be outfitted with polarizers that either transmitted or blocked the light. This would cause the keys, to the outside observer, to appear to have valid paths between holes with polarizations that would preclude light from the lock from entering them. For example, if the path from holes in the key corresponding to the locations of holes “a” and “b” in the reader included a filter that only passed light polarized in the “y” axis, the key would appear to have a connection from “a” to “b” to an unpolarized, or “y” polarized scanner, but not to an “x” polarized one. Inversely, polarizers could also be used to create paths between LEDs fitted with dissimilar filters. For example, if LED “a” were fitted with a “y” polarized filter, and LED “b” were fitted with an “x” polarized filter, an unfiltered optical fiber between the two would produce no useable signal path between LEDs “a” and “b,” however a fiber that rotated 90° inside the key (or included a polarizer oriented at a 45 degree angle between the “x” and “y” polarizations) would yield an optical path. The only real barriers to practicality of these techniques are the cost and attenuation factors of polarization preserving optical fibers, both of which are higher than their corresponding values for non-polarization-preserving fibers.

Another optical device that could be utilized for increased security is the beam splitter. Beam splitters could be employed to create directional variation in the network, dependent on their orientation. A beam splitter that takes light from optical port “a” and delivers it to optical ports “b” and “c” would not take light from either “b” or “c” and deliver it to the other two, but rather light entering the beam splitter from either of these optical ports would be directed to port “a” alone.

Movable, reconfigurable optics incorporated into the key could also be used as a powerful tool for security. In order to ensure the mechanical ruggedness of the keys and readers,

a fully sealed design with no external moving faces or electrical contacts would be preferred, so I would indicate magnetics as a prime choice for the actuation system. Inside the key, optical elements including pieces of optical fiber, beam-splitters, polarizers, and dichroic elements could be mounted on small magnetic sleds, loaded into guide-ways with weak springs. The springs would ensure that without an externally applied magnetic field, the optical elements on the sleds would not be aligned with their respective optical pathways. Inside the lock, fixed magnets aligned with the locations of the magnetic sleds in the lock's mating keys would move the elements into alignment upon key insertion. Furthermore, if the lock were constructed with electromagnets, multiple state movable optics could be used. An example of this would be a pair of dichroic elements, which in one position prevented green light from passing through a given optical fiber bundle, but with the application of an external magnetic field, shifted to deny red light passage. Polarizing filters are also a good candidate for this technique, as the 45 degree polarizer previously discussed could be moved into and out of alignment using the magnetically actuated sled.

Perhaps the simplest way to increase the number of combinations achievable with one key would be to allow for the routing of fibers between more than two optical ports inside the key. In a simple case of this, for a key with three holes in it, there are four combinations of optical port interconnection networks that leave no port disconnected from the system. For a four optical port key, this number jumps to 41 combinations, though not all of these combinations will be distinguishable from the reader's point of view. To contrast, a system that only allows optical ports to be paired with one other port allows for only three key patterns. An issue that must be taken into consideration with this implementation, however, is the fact that connecting multiple optical ports together cuts down on the light that can be passed between the ports in a

set amount of time. For example, in a four port key that allows each optical port to be connected to at least one, and at most two other ports would require at worst twice the read time, as any light entering an optical port connected to two other ports would immediately be split in half, necessitating a read time twice as long as in the one connection per optical port case in order for the LEDs acting as receivers to register light.

Now, while this might seem simple enough to fix by just multiplying the time the reader needs to read a singly connected key by the maximum fan out of any one optical port in the desired key, that is a losing strategy. Referring back to Tables 2 and 3 in Section 3.4, it is evident that this strategy will fail for a key with a maximum allowed fan out of more than twenty-five. As the minimum required exposure time for a singly connected key is $160\mu\text{s}$, and the maximum allowable exposure time is 4ms , a key with a maximum allowed fan out of more than 25 would require an exposure time of greater than 4ms , leading to some LED detectors reporting a false positive for light. This could be mitigated to some extent by machining with tighter tolerances, and placing the reader inside a completely dark enclosure, however in order to beat the parasitic device current due to thermal current and junction resistance the device would have to be actively cooled, which could prove to be a substantial nuisance. Additionally, going overboard with interconnections in the aforementioned style would lead to total scan completion times being multiplied by the maximum allowable fan out per optical port, as in order to verify the key, the reader would have to scan every optical port on the key for enough time to give an accurate reading of a port with the maximum allowed fan out.

6.0 Summary

The system outlined in this paper has the potential to become a new standard for the security industry. It is resistant to both malicious attacks, as well as day to day wear and tear, which in combination bring down most other key systems rather quickly. The keys are safe to use in hostile environments that would preclude the use of other types of keys. The purely passive nature of the keys leaves no energy signature to analyze, and the sheer combinatoric bulk of the optical options available obfuscates the inner workings of each key, making it an extremely costly system to defeat, time wise. Viability of the system as a product has been shown by the 60,000£ grant given to a system that is the forerunner to this work, and with the improvements put forth in this paper, the new system is even more secure.

7.0 Contributions

In this thesis, I made four main contributions.

1. I designed a key system with a lock that was able to probe the optical channels of a key bidirectionally, with minimal components.
2. I fabricated both the lock and key in aluminum, populated and installed the circuit board for the lock, and wrote the code to run it.
3. I wrote a test suite of code to troubleshoot various features of the system.
4. I came up with, and evaluated concepts for further improving the system in the future in order to enhance its security.

Appendix 1: Code

Unified makefile for all codes except those including serial UART:

```
# Fuses calculated @ http://www.engbedded.com/fusecalc/

DEVICE = atmega644p
CLOCK = 8000000
PROGRAMMER = -c usbtiny
OBJECTS = main.o
FUSES = -U lfuse:w:0x62:m -U hfuse:w:0x99:m -U efuse:w:0xff:m

AVRDUDE = avrdude $(PROGRAMMER) -p $(DEVICE)
COMPILE = avr-gcc -Wall -Os -DF_CPU=$(CLOCK) -mmcu=$(DEVICE)

# symbolic targets:
all: main.hex

.c.o:
    $(COMPILE) -c $< -o $@

flash: all
    $(AVRDUDE) -U flash:w:main.hex:i

fuse:
    $(AVRDUDE) $(FUSES)

install: flash fuse

clean:
    rm -f main.hex main.elf $(OBJECTS)

# file targets:
main.elf: $(OBJECTS)
    $(COMPILE) -o main.elf $(OBJECTS)

main.hex: main.elf
    rm -f main.hex
    avr-objcopy -j .text -j .data -O ihex main.elf main.hex
```

Revision 1, code designed to test the sensing capabilities of the LEDs:

```
#include <avr/io.h>
#include <util/delay.h>

int setup(void)
{
    MCUCR |= (1 << PUD); //disable pull up resistors
    DDRD = 0x7F; //the single input is the button on port d pin 7
    PORTD=0x00;
    DDRC=0xFF; //port c is all outputs
    PORTC=0x00;
}

int state=0x00; //state is just a storage variable for the state of the light sensing LED pins

void dly(int a){
    _delay_ms(a); //typing "dly" is faster than "delay"
}

int main(void)
{
    setup();
    while (1){ //loop forever
        DDRA=0xFF;
        PORTA=0xFF; //precharge the LEDs
        dly(100);
        if (PIND&0x80){ //if the button is pressed
            DDRA=0x00; //turn port a into inputs
            _delay_ms(10); //wait for the charges to decay
            state=PINA; //read port a
            if (state==0xFF){
                PORTD|=0x7F; //if the charges did not decay enough to be read as low
                values, blink the LEDs.
                dly(100);
                PORTD&=0x80;
            }
        }
    }
    return 0;
}
```

Revision 2 code, designed to check the ability of an optical fiber to couple light from one LED to another.

```
#include <avr/io.h>
#include <util/delay.h>

int setup(void)
{
    MCUCR |= (1 << PUD);
    DDRD=0x7F;
    PORTD=0x00;
    DDRC=0xFF;
    PORTC=0x00;
    return 0;
}

int state=0x00;

void dly(int a){
    _delay_ms(a);
}

int main(void)
{
    setup();
    while (1){
        DDRA=0xFF;    //junction precharge
        PORTA=0xFF;
        dly(10);
        if (PIND&0x80){    //wait for button press
            PORTA=0xFE;
            DDRA=0x01;    //switch to read for all LEDs but the emitter
            PORTC=0x01;
            _delay_ms(10);    //let charges decay
            state=PINA;    //record port state for testing
            if (((~PINA)&0b00000010)>0x00)&&((PINA&0b000000100)>0x00)){ /*if there's
                light on the LED coupled to the fiber, and no light on the LED
                not coupled to the fiber, give a green light on the status LED
                bank */
                PORTD|=0b00010000;
            }
        }
    }
}
```



```

        PORTD&=0b10011111;
    }
    if(((PINA&0b00000010)>0x00)&&((PINA&0b00000100)>0x00)){ /* if light is
        detected on both the LED coupled to the fiber, and the LED not
        coupled to the fiber, give a yellow light */
        PORTD|=0b00100000;
        PORTD&=0b10101111;
    }
    else{ // else give a red status light
        PORTD|=0b01000000;
        PORTD&=0b11001111;
    }
    _delay_ms(1000);    //hold the status light for a second for review
}
return 0;
}
}

```

Revision 3 code, designed to sequentially illuminate all eight sensing LEDs:

```
#include <avr/io.h>
#include <util/delay.h>

int setup(void)
{
    register unsigned char newMCUCR = MCUCR | _BV(JTD);
    MCUCR = newMCUCR;    //disable JTAG header
    MCUCR = newMCUCR;
    MCUCR |= (1 << PUD); //disable pull up resistors
    //MCUCR |= (1 << JTD);
    //MCUCR |= (1 << JTD);
    DDRD = 0x7F;        //port d all outputs save the button
    PORTD=0x00;
    DDRC=0xFF;         //ports c and a all outputs
    DDRA=0xFF;
}

int main(void)
{
    DDRA=0xFF;
    DDRC=0xFF;
    PORTA=0xFF;
    PORTC=0xFF;
    while(1){
        if (PIND&0x80){ //on button press, scan
            PORTC=0xFF;
            PORTA=(0b11111110);
            _delay_ms(10);
            PORTA=0b11111101;
            _delay_ms(10);
            PORTA=~0b00000100;
            _delay_ms(10);
            PORTA=~0b00001000;
            _delay_ms(10);
            PORTA=~0b00010000;
            _delay_ms(10);
            PORTA=~0b00100000;
```

```
        _delay_ms(10);
        PORTA=~0b01000000;
        _delay_ms(10);
        PORTA=~0b10000000;
        _delay_ms(10);
        PORTA=~0b00000000;
    }
    else{
        _delay_ms(10); //if no button press, wait
    }
}

return 0;
}
```

Revision 4 code, designed to scan a key, and compare it to an approved code database:

```
#include <avr/io.h>
#include <util/delay.h>

int state=0x00;
int map1=0x00;
int map2=0x00;
int map3=0x00;
int map4=0x00;
int map5=0x00;
int map6=0x00;
int map7=0x00;
int map8=0x00;

int setup(void){
    register unsigned char newMCUCR = MCUCR | _BV(JTD);
    MCUCR = newMCUCR;
    MCUCR = newMCUCR;
    MCUCR |= (1 << PUD);
    DDRD = 0x7F;
    PORTD=0x00;
    DDRC=0xFF;
    PORTC=0x01;
    DDRA=0xFF;
    PORTA=0xFE;
}

void map(void){
    //this function reads the LEDs, and stores the result in a
    //memory location dependent on which LED is currently lit

    if(state==0x00){
        map1 = PINA;
    }
    if(state==0x01){
        map2 = PINA;
    }
    if(state==0x02){
        map3 = PINA;
    }
    if(state==0x03){
```

```

        map4 = PINA;
    }
    if(state==0x04){
        map5 = PINA;
    }
    if(state==0x05){
        map6 = PINA;
    }
    if(state==0x06){
        map7 = PINA;
    }
    if(state==0x07){
        map8 = PINA;
    }
    return ;
}

```

```

void regurgitate(){
//this function displays the contents of the memory (filled by
//the last scan) on the LEDs to indicate what the reader saw.
    DDRA = 0xFF;
    PORTA = 0x00;
    PORTC = map1;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map2;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map3;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map4;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map5;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map6;

```

```

        _delay_ms(300);
        PORTC = 0x00;
        _delay_ms(10);
        PORTC = map7;
        _delay_ms(300);
        PORTC = 0x00;
        _delay_ms(10);
        PORTC = map8;
        _delay_ms(300);
        PORTC = 0x00;
    }

void checkKey(void){
    if((map1==(0x7E))&&(map2==(0xDD))&&(map3==(0xEB))&&(map4==(0xB7))&&(map5==(0xEB))
    &&(map6==(0xDD))&&(map7==(0xB7))&&(map8==(0x7E))){ //check it against known key structures
                                                here use some elifs or something to
                                                add more keys.

        PORTD&=0x00;
        PORTD|=0x10;           //blink the green led
        _delay_ms(1000);
        PORTD&=0xBF;
    }
    else{ //if the map of the scanned key does not match the map of an
        authorized key.

        PORTD&=0x00;
        PORTD|=0x40;           //blink the red led
        _delay_ms(1000);
        PORTD&=0xDF;
    }
}

void dly(int a){ //quicker to type than delay
    _delay_ms(a);
}

int main(void)
{
    setup();
    while(1){
        if (PIND&0x80){

```

```

while(state!=0x08){ //because the last bit shift in the iteration will run the one
                    //off of the end and make state 0x00, saving me a reset
                    //command later in the code.
    int litted = (1<<state);
    DDRA=0xFF;
    PORTA=(~litted);
    PORTC=litted;
    dly(100);
    DDRA={litted};
    _delay_us(10);
    map(); //move the state of the sensor (and the emitter led anywho)
           //into memory
    state++; //increment state before looping
}
state=0x00; //reset state
checkKey(); //check the key against the pre authenticated patterns
regurgitate();
}
else{
    _delay_ms(10);
}
}

return 0;
}

```

Makefile for codes using USART. This makefile does not set the clock divide by eight option, allowing the code to run faster. As a result, all of the delays for code using this makefile need to be eight times as large.

```
# Fuses calculated @ http://www.engbedded.com/fusecalc/
```

```
DEVICE = atmega644p
```

```
CLOCK = 8000000
```

```
PROGRAMMER = -c usbtiny
```

```
OBJECTS = main.o
```

```
FUSES = -U lfuse:w:0xde:m -U hfuse:w:0x99:m -U efuse:w:0xff:m
```

```
AVRDUDE = avrdude $(PROGRAMMER) -p $(DEVICE)
```

```
COMPILE = avr-gcc -Wall -Os -DF_CPU=$(CLOCK) -mmcu=$(DEVICE)
```

```
# symbolic targets:
```

```
all: main.hex
```

```
.c.o:
```

```
$(COMPILE) -c $< -o $@
```

```
flash: all
```

```
$(AVRDUDE) -U flash:w:main.hex:i
```

```
fuse:
```

```
$(AVRDUDE) $(FUSES)
```

```
install: flash fuse
```

```
clean:
```

```
rm -f main.hex main.elf $(OBJECTS)
```

```
# file targets:
```

```
main.elf: $(OBJECTS)
```

```
$(COMPILE) -o main.elf $(OBJECTS)
```

```
main.hex: main.elf
```

```
rm -f main.hex
```

```
avr-objcopy -j .text -j .data -O ihex main.elf main.hex
```


Revision 5 code, designed to test USART functionality.

```
#include <avr/io.h>
#define BAUD 9600
#include <util/delay.h>
#include <util/setbaud.h>

void uart_init(void) { // Atmel provided macro for setting up serial link.
    UBRROH = UBRRH_VALUE;
    UBRROL = UBRRL_VALUE;

    #if USE_2X
        UCSRA |= _BV(U2X0);
    #else
        UCSRA &= ~(_BV(U2X0));
    #endif

    UCSRC = _BV(UCSZ01) | _BV(UCSZ00); // 8-bit data
    UCSRB = _BV(RXEN0) | _BV(TXEN0); // Enable RX and TX
}

void uart_putc(char c) {
    loop_until_bit_is_set(UCSRA, UDRE0); // Wait until data register empty.
    UDR0 = c;
}

void uart_crlf(void){ //carriage return/line feed command using control
                    // characters.
    loop_until_bit_is_set(UCSRA, UDRE0); // Wait until data register empty.
    UDR0 = 0x0a;
    loop_until_bit_is_set(UCSRA, UDRE0); // Wait until data register empty.
    UDR0 = 0x0d;
}

extern void uart_putint(unsigned int i){
    unsigned char a,b,c;
    c= (i%10) | 0x30; //ones
    i/=10;
    b= (i%10) | 0x30; //tens
```

```

    i=i%100;
    a=(i/10) | 0x30; //hundreds
    uart_putc(a);
    uart_putc(b);
    uart_putc(c);
}

int setup(void){
    DDRA = 0xFF;
    DDRD = 0x7F;
    MCUCR |= (1 << PUD);
    uart_init();
    PORTD|=(1<<PD6);
    _delay_ms(100);
    PORTD&=~(1<<PD6));
}

int main(void){
    setup();
    while (1){
        if (PIND&0x80){
            int i=3;
            while (i>0){
                uart_putint(255);
                uart_crlf();
                PORTD|=(1<<PD6);
                _delay_ms(500);
                PORTD&=~(1<<PD6));
                i--;
            }
        }
    }
    return 0;
}

```

Revision 6 code, final revision.

```
//#define F_CPU 8000000UL
#include <avr/io.h>
#define BAUD 9600
#include <util/delay.h>
#include <util/setbaud.h>

void uart_init(void) {
    UBRROH = UBRRH_VALUE;
    UBRROL = UBRRL_VALUE;

    #if USE_2X
        UCSROA |= _BV(U2X0);
    #else
        UCSROA &= ~(_BV(U2X0));
    #endif

    UCSROC = _BV(UCSZ01) | _BV(UCSZ00); /* 8-bit data */
    UCSROB = _BV(RXEN0) | _BV(TXEN0); /* Enable RX and TX */
}

void uart_putc(char c) {
    loop_until_bit_is_set(UCSROA, UDRE0); /* Wait until data register empty. */
    UDR0 = c;
}

void uart_crlf(void) {
    loop_until_bit_is_set(UCSROA, UDRE0); /* Wait until data register empty. */
    UDR0 = 0x0a;
    loop_until_bit_is_set(UCSROA, UDRE0); /* Wait until data register empty. */
    UDR0 = 0x0d;
}

extern void uart_putint(unsigned int i)
{
    unsigned char h,t,o;
    o= (i%10) | 0x30;//ones
    i/=10;
    t= (i%10) | 0x30;//tens
    i=i%100;
    h=(i/10) | 0x30;//hundreds
    uart_putc(h);
    uart_putc(t);
    uart_putc(o);
}

int state=0x00;
```

```

int map1=0x00;
int map2=0x00;
int map3=0x00;
int map4=0x00;
int map5=0x00;
int map6=0x00;
int map7=0x00;
int map8=0x00;

int setup(void)
{
    register unsigned char newMCUCR = MCUCR | _BV(JTD);           //disable JTAG
    MCUCR = newMCUCR;
    MCUCR = newMCUCR;
    MCUCR |= (1 << PUD);           //disable pullup resistors
    uart_init();
    DDRD = 0x7F;
    PORTD=0x00;
    DDRC=0xFF;
    PORTC=0x01;
    DDRA=0xFF;
    PORTA=0xFE;
}

void map(void){           //record the key state for evaluation later.
    if(state==0x00){
        map1 = PINA;
    }
    if(state==0x01){
        map2 = PINA;
    }
    if(state==0x02){
        map3 = PINA;
    }
    if(state==0x03){
        map4 = PINA;
    }
    if(state==0x04){
        map5 = PINA;
    }
    if(state==0x05){
        map6 = PINA;
    }
    if(state==0x06){
        map7 = PINA;
    }
    if(state==0x07){
        map8 = PINA;
    }
}

```

```

    }
    return ;
}

void regurgitate(){           // a function for troubleshooting
    DDRA = 0xFF;
    PORTA = 0x00;
    PORTC = map1;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map2;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map3;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map4;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map5;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map6;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map7;
    _delay_ms(300);
    PORTC = 0x00;
    _delay_ms(10);
    PORTC = map8;
    _delay_ms(300);
    PORTC = 0x00;
}

void checkKey(void){
    if((map1==(0x7E))&&(map2==(0xDD))&&(map3==(0xEB))&&(map4==(0xB7))&&(map5==(0xEB))
        &&(map6==(0xDD))&&(map7==(0xB7))&&(map8==(0x7E))){ //check it against known
        key structures here use some elifs or something
        PORTD&=0x00;
        PORTD|=0x10;           //blink the green led
        uart_putc('U');       //transmit user 1 over serial
        uart_putc('s');
    }
}

```

```

        uart_putc('e');
        uart_putc('r');
        uart_putc(' ');
        uart_putc('1');
        uart_crlf();
        _delay_ms(1000);
        PORTD&=0xBF;
    }
    if((map1==(0xED))&&(map2==(0xD7))&&(map3==(0xBB))&&(map4==(0x7E))&&(map5==(0x7E))
        &&(map6==(0xED))&&(map7==(0xBB))&&(map8==(0xD7))) { //check it against known
        key structures here use some elifs or something
        PORTD&=0x00;
        PORTD|=0x10; //blink the green led
        uart_putc('U'); //transmit user 2 over serial
        uart_putc('s');
        uart_putc('e');
        uart_putc('r');
        uart_putc(' ');
        uart_putc('2');
        uart_crlf();
        _delay_ms(1000);
        PORTD&=0xBF;
    }
    else{
        PORTD&=0x00;
        PORTD|=0x40; //blink the red led
        _delay_ms(1000);
        PORTD&=0xDF;
    }
}

void dly(int a){
    _delay_ms(a);
}

int main(void)
{
    setup();
    while(1){
        if (PIND&0x80){
            while(state!=0x08){ //because the last bit shift in the iteration will run the one
                off of the end and make state 0x00
                int litted = (1<<state);
                DDRA=0xFF;
                PORTA=~litted;
                PORTC=litted;
                dly(50);
                DDRA=(litted);
            }
        }
    }
}

```

```
        _delay_us(400);
        map(); //move the state of the sensor (and the emitter led anywho)
into memory
        state++; //increment state before looping
    }
    state=0x00; //reset state
    checkKey(); //check the key against the pre authenticated patterns
    //regurgitate();
}
else{
    _delay_ms(10);
}
}
return 0;
}
```

¹ Benjamin Laxton, Kai wang, Stefan Savage. "Reconsidering Physical Key Secrecy: Teleduplication via Optical Decoding." ACM CCS 2008, Alexandria, VA, 2008

² Russel Ryan, Zack Anderson, Alessandro Chiesa. "Anatomy of a Subway Hack." 2008.
http://tech.mit.edu/V128/N30/subway/Defcon_Presentation.pdf

³ Hart, Robert Lee. "Opto-Electronic Lock Device." Patent 4,090,175. 16 May 1978.

⁴ Rice-McDonald, Jeremy Christopher Doune. "Security System." Patent EP 1,272,721 B1

⁵ <http://en.wikipedia.org/wiki/File:Optical-fibre.svg>

⁶ Anchor Optics. 2014. <http://www.anchoroptics.com/catalog/product.cfm?id=354>

⁷ S. Shibata, S. Mitachi, S. Takahashi. "High Numerical Aperture Multicomponent Glass Fiber." Applied Optics 19,9. 1980.

⁸ Bolesh J. Skutnik, Brian Foley, Kelly B. Moran. "High Numerical Aperture Silica Core Fibers." Society of Photo-Optical Instrumentation Engineers. 2004.

⁹ Corning Cable Systems. "Minimum Optical Fiber Bend Radius" AEN 21, Revision 6. 2006.