

Sensors & Security: Summer Term II

30 June – 8 August 2003

Thomas Rawley
Stephen Verdi
Mike Voelker

Contents

Access Control	2
<i>Abstract</i>	2
<i>Research</i>	2
<i>The Bioscrypt System</i>	4
<i>Reliability and Robustness</i>	7
<i>Cost Analysis</i>	8
<i>Installation Phase & Timeline</i>	8
<i>Contacts</i>	8
<i>Conclusion</i>	9
The Presence Sensor	10
<i>Abstract</i>	10
<i>Research and Development</i>	10
<i>Setup</i>	12
<i>The Timing Circuit</i>	12
<i>Walkway Layout</i>	14
<i>Reliability and Robustness</i>	15
<i>Cost Analysis and Payback</i>	15
<i>Installation Phase & Timeline</i>	16
<i>Future Advancement</i>	16
<i>Conclusion</i>	17
Appendix A	18
Appendix B	19
Appendix C	20

Access Control

Abstract

The Sensors & Security team spent a great deal of the first summer term researching the multiple ways by which access into the DELTA Smart House may be controlled. Today's most common way of restricting access to a home is the simple - yet reliable - lock-and-key. Though this system has proven very effective, we recognized access control as an opportunity to move forward and integrate modern technology to improve this old-fashioned mechanism. A person who protects their home with a standard lock is burdened by the need to carry a key whenever he or she leaves the home. If a key is lost, often times locks must be changed or new keys must be made. Keeping these downsides in mind and brainstorming a few innovative ideas of our own, we compiled a short list of features and requirements which we foresaw as integral parts of our access control system. We decided the locking system must be:

- Unsurpassable: there will be no means by which intruders are able to break in
- Selective: only those with permission to enter the DELTA house are able to do so, and a log is kept of who has entered
- Convenient: users should not need to carry anything with them to be capable of unlocking the house
- Modern: uses new and emerging technologies
- Expansive: allows a large number of users stored in a database
- Customizable: different users can set unique preferences to trigger upon entrance

A house that has an access control system possessing these qualities will provide utmost security and comfort for its residents.

Research

In our research we found a variety of options for locks, however very few of them met all of our requirements. The first system we came across was the iLock, made by Creative Control Concepts. The details of the iLock system are described in a previous report on the security system. iLock was ruled out (at least for primary access to the home), because it required that users carry a magnetic "key" with them at all times. Also, the system was designed to allow a mere seven users, and modification of this would be quite long and tedious. Another idea was to use infrared-controlled locks. This is the same keyless entry system used in many vehicles. Again, if we were to implement this system, members would have to carry a bulky keychain. There also would be no way to distinguish one user from another in order to keep a log. The third system, a keypad, would satisfy most of the listed requirements. One problem with a keypad is that codes can be easily passed around. When a key is involved, only a certain number of copies are made, so it is possible to keep track of all of them. With a numeric code, however, it is impossible to keep track of who knows it and to who they have passed it along. This, obviously, raises security concerns. Another

concern is that the keypad does not offer the cutting-edge technology that we seek. Keypads have been around for years, and are already quite commonplace.

After sorting through all of our options, we found the perfect fit to our requirements in biometrics. Biometrics use the unique features of the human body to identify a person. The three most common biometric systems are fingerprint scanners, iris scanners, and facial patterning units. In-depth internet searches were performed on all three of these systems. Similar problems were found with both iris scanners and facial patterning units. First, and perhaps most important, neither of these techniques seems well enough developed for use in the home. There are advanced versions of these systems which are currently in use, but generally we did not see them put to use in residential sites. Obviously it is crucial that our system function properly and that it remain within our price-range. Because fingerprint readers are currently being designed for homes and small businesses, they are lower in cost than other biometrics, which are more often seen used for government or large corporation use. In terms of the iris scanner, we were also slightly concerned about its convenience. These require that users peer into their sensors while the eyes are thoroughly scanned. Since the unit must be placed beside the door, it might be inconvenient for users to have to put their face right up to the wall and hold it steady for a moment. Fingerprint readers are better in this sense – one must only reach over to the reader with an arm, rather than move the entire body. It is also a quicker scanning process.

Iris scanners and facial patterning systems ought not be discounted as possibilities. As these systems are further developed in the future, they might make an interesting, cutting-edge addition to the access control system. However, for our purposes at this point in time, fingerprint readers provide the best solution to our access control concerns.

Once fingerprint readers were decided upon, we looked back at our research of biometrics and sorted through the various fingerprint reader developers. At Georgia Tech, fingerprint readers were used for the access control system to their own technology-based house, called The Aware Home. They used a reader designed by DigitalPersona that was meant to be used for logging into a computer. We e-mailed each of the three members of the Aware Home Fingerprint team, and got positive feedback on the DigitalPersona's performance from each of them. Our main concern here was that the reader used was not initially developed to monitor physical access control. This meant that we would be required to develop our own system (hardware and software) which would adapt the reader for our purposes. The Aware Home members wrote their own programs to allow the reader to serve as a door controller, and they offered to send a copy of the program to us if they were able to find it. In the midst of conversation with the Aware Home team, we decided to try to get in touch with the person in charge of installing fingerprint readers around Duke's campus. It was brought to our attention that an article had been published in The Chronicle several months ago regarding the readers around campus, and we thought we might find the name of the person who installed them for Duke. After searching The Chronicle archives online, we found the article and the name of Duke's head locksmith, Jimmy Tilley. We arranged a meeting with Tilley during which he showed us a fingerprint reader and how they can be used, and spoke to us about other new-age access control systems. The reader he uses around campus is one made by Bioscrypt and distributed by Integrated Biometric Tech out of Nashville, Tennessee. We were very impressed with this reader and the ease of use. After the meeting, Tilley contacted the contractor at Integrated Biometric Tech and informed him of our project and then passed on the contact information to me. We called the company and were forwarded to Richard Hutchinson, who immediately agreed to send us the entire package necessary to control and monitor a door for us to use for a limited time. We decided to halt further research into other systems until we tested that which would now be sent.

The Bioscrypt System

The package sent by Hutchinson included a VPass fingerprint reader, fingerprint enrollment software, a door controller with corresponding software, an electronic doorstrike, and a 120 VAC to 12 VDC power transformer. Bioscrypt's V-Pass model is exactly what we would like to use in the house – a simple stand-alone reader. Other models have fingerprint and prox-card readers or fingerprint readers and keypads. These units are too bulky and, for our purposes, probably unnecessary.

Setup of the entire system is quite simple once the correct cables and connectors are obtained. With the system came a cat5 cable which carried a signal over RS-232. This cable was terminated on one end with an RJ-11 connector and on the other end by a serial connector. The serial connector was to be plugged into the computer and the RJ-11 connector was plugged into the base of the reader.



This cable is necessary only for the enrollment of new fingerprints into the system. Also in order to enroll new fingerprints, the VeriAdmin software was necessary. This software came in our package as well. Once installed, we were able to enroll fingerprints very quickly and easily. One point should be enforced with regard to the VeriAdmin software:

VeriAdmin is able to create a network of fingerprint readers for those users who need multiple readers in a building. For this reason, the software must have some means of knowing one reader from another. The burden of assigning IDs is placed on the user. The ID numbers for each reader is called its Transmit ID. When installed on a computer, the VeriAdmin software creates a file called `unitids.dat`. This file contains the default Transmit IDs, listed 1 through 9. When using a V-Pass, however, this file must be edited. In order for the software to recognize the reader as a V-Pass, the unit ID must be a number followed by a colon and the letter M. Thus, in my case, I deleted the numbers 1 through 9 in the file and replaced them with only 1:M. This should be repeated depending on the number of readers planned to be used in the network. So, for example, if I were to use another reader in the house, I might like to network the two. Then I would need to open `unitids.dat` again, and add 2:M following the 1:M already added previously.

With VeriAdmin, any finger can be enrolled into the system. Before enrolling, a template ID number must be assigned to the fingerprint about to be recorded. As far as we know, this can be any number of up to 10 digits; however, we did have problems when many digits were used. Once this is done and the fingerprint is recorded, the record created can either be saved to the PC or to the reader itself. If a network of readers is created, enrollment must only be performed at one of the readers, and then this record can be broadcast to all of the remaining readers.

The next step is to get MyEasinet working. MyEasinet is a door controller system consisting of both hardware and software. The hardware connects directly to the reader through cat5 cable. Through this series of wire a signal is sent when the reader recognizes a fingerprint as it is scanned. The door controller is also connected to a computer through a COM port. We found it necessary to purchase a 25-pin to 9-pin

adapter in order to fit this connector into our COM port. The MyEasinet software must be installed on the computer. This software logs all events and users. Thus, when a user is accepted by the reader, the signal is sent to the door controller which tells the software who is at the door and at what time. If the software recognizes the person as having access to the door at that time, it then triggers a relay within the door controller to flip, which then causes a door strike to unlock. Any functions desired for the door controller can be set within the MyEasinet software. When new users and door controllers are added to this software, there are a few critical points to keep in mind:

MyEasinet is able to recognize who is entering by reading the signal sent by VeriAdmin. VeriAdmin tells MyEasinet which template ID has been accepted. MyEasinet must then match this number with a user's name from its database. To do this, when a new user is added, he must be given a card number which is identical to his template ID number. Again we believe that it should work with any number of 10 digits or less, however, when tested, it did not function properly with the larger numbers. Also, the default setting for this software is to grant new users no access to any doors at any time. One should be sure to change this setting as desired.

Another important setting in the MyEasinet must be changed when a new door is added. For the Bioscrypt VPass readers, the door controller must be told that these are "Weigand Readers" capable of accepting "Card, Pin, or Code."

Once all of these steps are taken, the access control system should be set and ready to operate as intended by its manufacturers. We were able to set it up and run it all without a problem, however we decided that we wished to customize the system a bit more. There were essentially two main ideas we wished to incorporate into this fingerprint system. First, we wanted to be able to view the log of users from the internet. This would be both a safety feature and a "show-off" feature, where we could demonstrate to people that the house is capable of monitoring and reporting such information. The second idea was to have the house greet a resident as he enters the house. Though these ideas might sound simple to carry out, they actually required some clever programming, done by Thomas. He describes his work first on the posting of the database on the internet, and then on the personalized greeting as one enters the house:

Internet Database:

This project was conceived and implemented after the fingerprint scanner was received. It was brought to my attention that we may want to create a web page that, when queried, will show a list of people who have entered the house and some accompanying information.

After researching this project for a day or two, I found a way to do this. Specifically, the fingerprint scanner software (MyEasinet) writes all events (along with a ton of other information not pertinent to this particular project) to a database named MyEasinetSystem.mdb. This Microsoft Access Database was the key to this project. After figuring out that the events were recorded in this database dynamically (as someone enters) the only thing left to do was to create a program that parsed this database and converted the information in it into a format that a web browser can read. At this point in time, that format is HTML. In the future, I might look into changing this format to XML and have the browser format this XML file containing all the events via an XSLT script

or CSS file. This has one main advantage. If we have the information in an XML file, we can use it for other purposes in the future.

I weighed the pros and cons of the different programming languages to use in an effort to realize the above goal. My choice was Perl for its simplicity in code and how easy it was to obtain modules for accessing the Microsoft Access database. I created a simple Perl script that in essence, took the information from the event log and converted it into HTML. You will find a copy of this code in Appendix A. As you can see, it is very short and easy to understand.

One of the problems we may run into in the future is being able to actually access this information from the internet. Right now, everything is done on the local computer. This works flawlessly. Since the data is contained in an Access database, Windows seems the likely choice for the server which will handle requests to view this event log. It would make life much easier if the server was Windows based. It is much harder to access an Access database in Unix, although I believe it can be done. Once this issue is resolved, I think it is a simple matter to get the Perl script up and running on the internet.

As stated at the beginning and as is most likely evident now, we need to be careful and choose what information is displayed on the internet. We do not want to violate people's privacy by publishing when they entered and left their own house. But, there are ways to avoid doing this, which can be discussed at a later date.

Personalized greeting:

First, lets take a step back and get an idea of how the home automation software (HAL) actually recognizes that someone has entered the house. We needed an electronic door lock, which was provided to us by Integrated Biometric Tech. The technical premise behind this is very simple. When the door lock has been given power (the door lock opens) the Ocelot processes that event through the use of a relay and the supervised input mode on the Ocelot itself. When this occurs, the Ocelot sends an ASCII message via the serial cable back to the computer telling HAL to turn on a sensor. HAL is listening on this COMM port and there is a sensor in HAL that detects the data being sent through the COMM port and this sensor then turns on (if the correct ASCII message was sent).

Once this sensor turns on, we have a rule in HAL that runs a simple Perl script. The Perl script opens the fingerprint database and extracts the most recent event (which is the second to last row in the Events table of the MyEasinetSystem database). The script takes the card number from this row and cross references this number with another database. This second database stores the card number and name for each person who is in the fingerprint database. It is named ID_Table and as of right now, this database has to be manually compiled. In the future, I believe it is possible to have only the MyEasinetSystem database and we can extract all the information from it.

So now that we have a name, we can personalize the greeting. My first idea was to have a flag in HAL for each person in the house who wants a personalized greeting. The Perl

script would then, using the HAL_Interface.exe program, change the correct flag in HAL to true. Then, via a rule in HAL, HAL would text to speech a predefined sentence welcoming the user into the house.

Although this approach works well and accomplishes the original goal of personalizing a greeting, it is messy. There is one sensor and the same number of flags as people living in the house. This makes it very confusing to program HAL and clutters up the rules. So I felt there had to be a better way to accomplish this goal without all the clutter.

After some brainstorming, I came up with an idea that I thought would work. Everything was the same up to the Perl script. The script read in the databases the same and got the same name out of them, but instead of using the HAL_Interface.exe program to change a specific flag in HAL, the script wrote a sentence out to a file. This sentence is what HAL reads and includes the generic greeting followed by the personalized greeting, such as "Welcome home Thomas." The script then tells HAL to read this file and text to speech it via the HAL_Interface.exe program. This allows there to be only one sensor and one flag in HAL.

Utilizing the second approach has a number of benefits. First, it gets rid of all that clutter in HAL and makes the rules section easier to read and understand. Second, as a side effect of the way this implementation works, if for some reason the persons name is not the in ID_Table database or something goes wrong, the Perl script still writes the generic greeting to the file. For instance, if a name was not found, the file would contain "Welcome home" instead of "Welcome home Thomas." Then HAL reads the file. So, if a name was not found, a greeting will still be announced over the speakers.

Future expansion of this system may include only having HAL welcome you home over the entrance hallway speakers, so that everyone else in the house does not have to hear the message. In addition, I would like to use only one database to extract a name. This is possible, but I don't know how efficient it is. All the information I need is stored in the MyEasinetSystem database. It is just in a couple of different places. Until I have a better understanding of SQL and all the commands possible, I am going to leave the program like it is.

After these advancements, we are very pleased with the final product, and believe this system is ready to be installed in the Smart Home.

Reliability and Robustness

As opposed to many of the projects undertaken during this design phase, the fingerprint reader and corresponding access control system has been one which has proven almost perfectly reliable. There have been no occurrences of acceptance of a foreign fingerprint, nor has it denied a familiar fingerprint when placed correctly on the scanner. This is quite comforting, for access control is one system which we truly cannot afford to malfunction.

The reader must be adequately durable to resist wear from finger placement and to function through all weather conditions. The sensor on which fingers are placed is designed to function with great accuracy even after years of repeated use. To protect the sensor from weather, a hood can be added to the mounting structure for the reader. This will keep rain and snow from accumulating on the device. In addition, the reader is designed to operate within temperature limits that are well beyond the range in Durham.

Cost Analysis

Integrated Biometric Tech has been generous enough to lend us this entire system free of charge. The only drawback is that we must return it once testing is complete. I am currently working on a request for a donation from them.

If we are to not receive the donation, I believe the entire system will cost about \$1250 for a single reader. If we were to add another reader, it would be significantly cheaper than the first, because our door controller can accept two readers. Thus we would only need to purchase another V-Pass.

Installation Phase & Timeline

The fingerprint system is currently defined as a Phase 3 project, however access control is defined as Phase 1. This is because fingerprints were merely an idea at the time that phases were defined. After thoroughly developing the fingerprint system, I am confident in saying that this can serve as our access control, and no other system need be installed prior to it. The house should be wired for the readers (most likely one at the front door and another at the back, though there may also be use for one to grant access into the control room) and the door controllers as the house is constructed. This can be setup to function starting the first day the house is habitable. The only thing hindering the immediate use of this system is the time we must reserve for enrolling all necessary fingerprints into the system.

Contacts

James “Jimmy” Tilley

Locksmith Supervisor – Duke University
200 Facilities Ctr
Box 90160
660-4223
jimmy.tilley@duke.edu

Richard Hutchinson

Integrated Biometric Tech
25 Century Blvd. Suite 210
One Lakeview Place
Nashville, TN 37214
1-888- 245 -1114

Conclusion

Our team was engaged in thorough research for weeks trying to identify the perfect access control system to satisfy all of our demands. We encountered numerous possibilities, but only one was the perfect match: fingerprint scanners. Fingerprint scanners are the best option for us right now, offering versatility, security, cost-effectiveness, reliability, customizability, and cutting-edge technology. The system is also very user-friendly. With only limited help from our distributor, we were able to install and run the entire system without a glitch. We were even able to go so far as to customize the system to meet specific goals laid out for ourselves. After thorough testing, this system has proven itself as a reliable and viable addition to the Smart House security system. We have had the time to understand all of its capabilities and are prepared to install the system so that it is up and running as soon as the house is completed.

The Presence Sensor

Abstract

Many common homes use in-ground walkway lights to illuminate the pathway to their front doors. A major problem with these lights, however, is the amount of energy one may waste by using them. Generally, exterior lights are turned on at dusk and left on for much of the night, regardless of whether residents are expecting anyone to walk down the pathway or not. In the Smart House a major goal is energy efficiency. If we could think of a clever means by which the front walkway will illuminate only when it is in use, then we can eliminate much of the inefficiency of such lights.

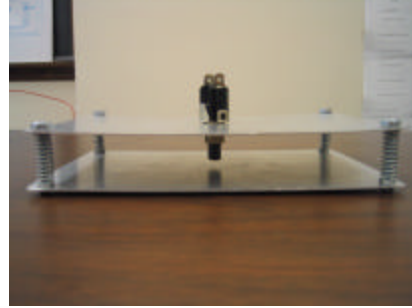
Research and Development

The heart of this project was the ability of the system to recognize whether or not the walkway was in use. There are several ways in which we could have done this. One method would be to use infrared motion detectors to sense movement at the walkway. This idea was attractive because it would require little research and original design. Motion detectors are commonly used to switch on household devices such as lights or cameras, and we could very easily arrange this system to function as we need it to. A motion-based system has several shortcomings as well, however. First, the sensors would have to be placed on the exterior of the house or somewhere else where they are in plain view of the walkway so that they can detect a person. We imagined this system would be quite unsightly, especially considering the numerous sensors that would be necessary for monitoring the entire walkway. Another problem with motion detectors would be false triggers. We would not want the lights to turn on each time an animal ran by or a nearby plant moved with the wind.

An idea that we settled upon was to use pressure sensors to recognize a footstep on the walkway. This would be a better idea than the motion detectors, because each pressure sensor could be hidden beneath the walkway stepping stones. This is where our primary research began. A sensor must be found that is able to be calibrated so that the stone alone would not trip the sensor, nor would an additional light weight (such as that of an animal).

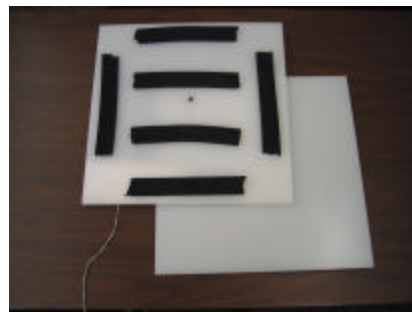
The first sensor we came across was called a load cell. Load cells are commonly used at truck weight stations and in many factory settings. They are constructed of a strain gauge imbedded in a hard metal or concrete. The strain gauge is simply a metal wire through which a current is sent. When the cell is compressed or stretched, the resistance of the wire is altered due to its change in shape. The change in resistance changes the current that is able to flow through the wire. This change in current can be detected and a weight is calculated based on this change. We were able to rule out load cells based on the size of the product. Load cells vary greatly in size, however we were unable to find a suitable small sensor which would be able to handle several hundred pounds of pressure. Also, load cells in general seemed a bit more complex than was necessary for our type of pressure sensing. We do not need to measure weight, only recognize pressure.

At this point, we decided to try to design our own sensor. The general idea was to build some model where a circuit was completed when pressure was applied. The first design (shown to the right) was simply two metal plates connected at the four corners by bolts surrounded by springs. A push-button switch was punched through the top plate. When pressure was applied to the top plate, it was depressed enough so that the switch was pressed. A circuit including this switch, a power supply, and an LED was all we needed for this to function properly. Obviously this was



only a rough prototype, however we could immediately see a great problem with this model: it would be very difficult to make on a small scale. Switches small enough to place beneath a stepping stone would be both difficult to find and most likely not durable enough to withstand hundreds of pounds of pressure.

The next design was created with size in consideration. Rather than a switch, we decided to create a sensor that was based on metal contacts. The metal contacts can be paper thin, and will serve much the same purpose as the switch: to complete our circuit. We used two plastic sheets with bolts screwed through holes in the middle of each. Then, strips of foam were glued to one of the sheets. When the two sheets were stacked, they were separated by a small distance created by the foam strips. With pressure, this foam would compress, allowing the two bolts to come in contact.



Using the same circuit as in our push-button switch model, we were able to make an LED glow when the top sheet of plastic was stepped upon. We were happier with this design than the earlier push-button setup, however we ran into a big problem with this design as well. Since this sensor will be located outside, it must be waterproof. In our case, water could easily flow between the plates and could potentially conduct electricity. If this were to happen, the water would essentially be completing the circuit, thus causing the lights to turn on.

As we were brainstorming a method of weatherproofing our latest design, we came across a strain gauge sold on SmartHome.com. As mentioned previously, strain gauges are thin metal wires which change resistance when their shape changes. Thus, the amount of current flowing through the wire will be altered by the resistance change. This gauge made by SmartHome.com is meant to be epoxied to the underside of floorboards. When a floorboard is stepped on, it experiences some minor flexing, which is enough to change the shape (and thus resistance) of the strain gauge. The sensor also comes with a processor which is able to recognize the change in current flowing through the gauge. We imagined creating a waterproof box and gluing this sensor to the lid of the box. This box could then be buried underneath a stepping stone. Pressure on the stone would be transferred to the lid of the box which would then flex in response. The gauge would report this flexing to the processor which would then send an output to turn on a circuit with our walkway lights. We decided this was the best idea at this point. We tended to have a bit more faith in the product than in our own since it was professionally fabricated, so we went ahead and purchased the sensor (called a *presence sensor*) and the corresponding one-zone processor (called the *111 processor*). There is also the option of purchasing a multi-zone processor which would allow three sensors to be monitored by one processor.

Setup

When we received the presence sensor and processor, we immediately epoxied the sensor onto a sheet of plastic which would eventually serve as the lid to our box. The sides of the box were constructed with wood 2 X 4s.

After thoroughly reading and understanding the directions sent with the processor and sensor, we were ready to take on the feat of setting it up for the first time. There are eight contacts on the processor (in order from left to right): A, C, B, NC, C, NO, +12, and GND. We will tackle one at a time.

There were four 1 M Ω resistors sent with the processor. These resistors are supposed to represent presence sensors that are not in the circuit and can be thought of as “false” sensors. The theory behind this is that the actual sensor changes resistance when it is bent and these resistors do not. For example, since we only have one presence sensor right now, we replace one of the resistors with the sensor itself (more later on how to replace the resistor). The first step was to wire two of the resistors in series and connect them between contacts A and C. These represent the two “false” sensors. The next step was to wire the other two resistors in series and place them between contacts C and B. But since we want to have a sensor in this circuit, we simply substitute the two wires on the sensor for one of the resistors. So, in actuality, we place one resistor and one sensor in series and place them between C and B.

The +12 and GND contacts are pretty self explanatory. You connect the positive lead of a 12 V power supply to the +12 contact and the negative lead of the power supply to the GND contact.

Now back to the remaining three contacts: NC, C, and NO. The C here is for Common and is connected via a jumper cable directly to the GND contact. This leaves the NC and NO contacts. The NC contact stands for Normally Closed and the NO contact stands for Normally Open. This is in reference to relay outputs. The NC output is closed (to ground) until the sensor is tripped and at that point, the output is disconnected so there is an open circuit. With the NO output, there is an open circuit until the sensor is tripped and then the output pulls to ground when the sensor is tripped. For instance, to power a relay you could connect one contact to NO and the other to +12 and the relay would close when the sensor is tripped.

Above, we have discussed what happens when the sensor trips. We also have the option to control the sensitivity of the processor so that it takes more strain on the sensor before the processor “detects” the presence of someone or something. This is useful for one main reason. We do not want the walkway lights to turn on when an animal walks across the sensor so we can decrease the sensitivity, thereby requiring more weight on the stone to trip the circuit. The lowest sensitivity gives us a threshold of about 40 pounds.

The Timing Circuit

In this section of the report on the presence sensor, we explain the technical aspects of the timing circuit we created in an effort to shed light on and otherwise simplify the complicated looking circuit. When we finally got the 111 processor working and understood its complexities, we decided we wanted a light to stay on for an extended period of time instead of only coming on when the processor was resetting itself. Therefore, we needed some sort of timer. We looked up and down for a prefabricated timer that would accomplish our goal, but to our dismay, could not find one. This forced us to create our own timer circuit.

Now that you know the reason for the circuit, the following explains how we built the circuit from beginning to end.

We needed an IC chip to create this circuit and we choose the 555 Timer chip for its simplicity and availability. You will find the pin diagram for the 555 Timer in Figure 1.

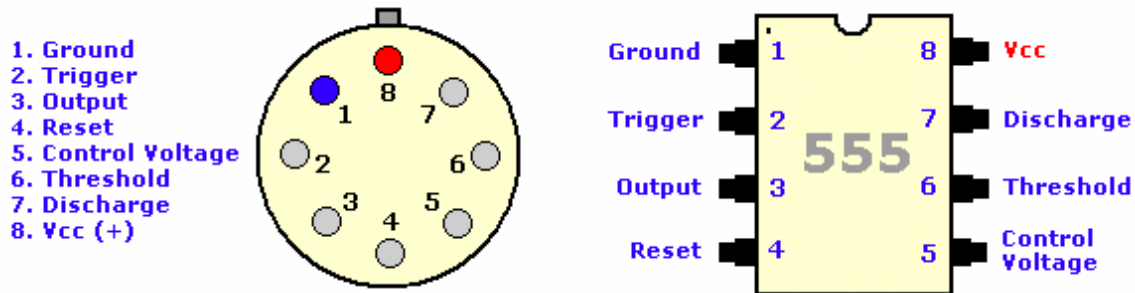


Figure 1: 555 Timer Pin Designations¹

We chose to use the 8-pin V package chip. The next step was to figure out how to integrate this chip into a circuit that would control how long our light was to stay on. We resorted to the internet for a sample circuit and found one that looked promising. It is shown in Figure 2.

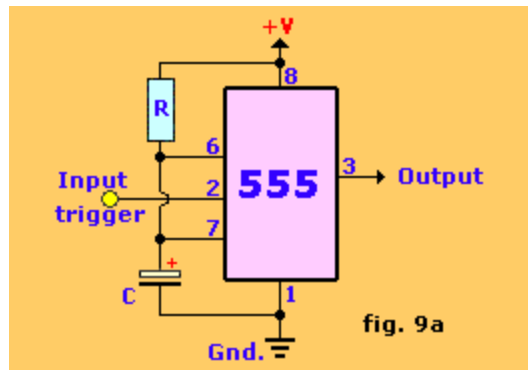


Figure 2: Timing Circuit

The resistor (R) and the capacitor (C) are connected in series between +Vcc and ground. This RC circuit is the timing portion of the larger circuit. To get the amount of time the output at pin 3 is high, you multiply R by C (which is the RC time constant). For instance, if we had a 1 M Ω resistor and a 1 μ F capacitor, the time the output at pin 3 is high is 1 second (R*C).

We took to the lab and attempted to construct this circuit. The above circuit would light up a small incandescent bulb for the amount of time specified by R and C. All we had to do was attach the coil of a relay from ground to the output and then attach our 120 V light bulb to the contacts on the relay. This however, was not so easy. In fact, it took us about two weeks to complete the timing circuit.

¹ Image by Tony van Roon acquired from: <http://www.uoguelph.ca/~antoon/gadgets/555/555.html>

To start with, we had to interface the circuit to the 111 processor from above. The problem is that when the NO or NC relay output of the processor trips, it connects to ground and not to +Vcc. This is a problem because the trigger input on the 555 Timer must remain high and when it drops below a certain level (2/3 of the control voltage at pin 5 which is the same as +Vcc) the output goes high. Since the NO or NC outputs pull to ground, the voltage never changes. Therefore we had to figure out a way to remedy this problem. After talking with Dr. George about this, he recommended a 1M Ω pull-up resistor to be put between the trigger input and +Vcc. We went back to the lab and tried this and it worked perfectly. However, soon we encountered more problems.

Now that the 111 processor and the timing circuit are interfaced, we have to deal with bounce. Bounce is the momentary voltage spike when a switch or relay closes or opens. This bounce causes the timing circuit to restart after it completes its cycle because when it tries to close the switch in the relay, it gets this bounce. To account for the bounce, we added a 1 μ F capacitor between ground and pin 5 (the control pin). This helped with the bounce we were getting.

In addition, to make the circuit more robust, we connected the reset pin (pin 4) and +Vcc. The circuit will operate fine without this connection, but just makes it more robust. With all these additions, we could turn on a small incandescent bulb just fine because it did not draw a whole lot of current. However, on the other hand, the relay coil drew more current than the output of the timer could support. We blew quite a few 555 Timers until we figured this out. To remedy this problem, we had to go again to Dr. George and he said to use a transistor. So we got a 2N3414 NPN transistor and put it into our circuit. The base of the transistor was connected to the output of the 555 Timer. The collector of the transistor was connected to +Vcc and the emitter was connected to a contact on the coil of the relay. The other contact of the relay coil was connected to ground. Using the transistor, the circuit provided enough current to be able to operate the relay properly.

Next we added the 120 V light to the contact side of the relay. We did this and once again ran into bounce. This time it was from the 120 V light. To remedy this problem, we had to add a 1 μ F capacitor across the contacts on the relay, specifically the two contacts that the light is connected to. This fixed that problem and after extensive testing, we got the thing working and put it in its final box. Hopefully, this circuit will come in handy in the future.

Once this circuit was operating as desired, we planned for its encasement. We wanted the entire system to be contained in one box, with the presence sensor mounted to the inside of the lid. We found a waterproof conduit box of adequate size and mounted all of the circuitry on a plastic rail inside. The box will be buried underneath a stepping stone with only the lid above ground. This way, if we have any problems with the sensors, we can simply lift a stone and remove the lid.

***See Appendix B for a schematic diagram of the presence sensor

Walkway Layout

Quite a bit of thought was put into the actual layout of the front walkway for the Smart House and the locations of our presence sensors. There are two main considerations which must be kept in mind with regard to lighting this pathway. First of all, each sensor, when tripped, will trigger at least one light ahead

of the walker as well as one light behind the walker. With this arrangement, the sensor network will work regardless of which direction a walker is moving. Second, each light can be connected to only one relay. If we were to put one light on several relays, we may cause an overload at the lamp. This would occur because each relay would complete a 120 VAC circuit which included the lamp. Thus the lamp might become part of several 120 VAC circuits, each of which has current flowing through and to the lamp simultaneously. With these requirements in mind, we designed the network drawn in Figure 3 below. In this drawing, a colored stone designates a stone with a sensor beneath it, and each stone will trigger on each lamp (designated by an asterisk) of matching color. For instance, stepping on the blue coded stone will trigger the two blue lights (blue asterisks) to turn on. Also note that the final stone (colored pink) will also trigger the front porch lights to turn on.

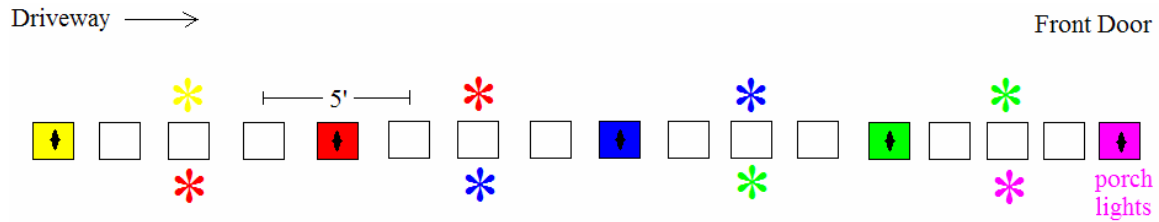


Figure 3: Walkway diagram

Reliability and Robustness

This project unfortunately still has questionable reliability. Upon our completion of the timing circuit, we moved all of the elements from our testing breadboard onto a standard PCI board. When soldering was finished and the circuit was tested, it no longer functioned properly. All aspects of the circuit were thoroughly checked for problems, but none could be identified. Ultimately we ended up completely rebuilding the circuit on another smaller breadboard. When connected on this new breadboard, it again worked without flaw. Confused by this, we decided to leave everything connected to the breadboard at this point. We feel that when it comes time for this circuit to be put to real use, we might like to try having our PCI board custom built, so as to avoid human error in the completion of this circuit.

The conduit box which contains our sensor is adequately strong. It has been tested with pressure and does not appear to show signs of weakening. It is also completely watertight. For these two reasons we do not anticipate any structural problems in the current plan for containment of this project.

Cost Analysis and Payback

In total, this is an inexpensive project. The costs are as follows:

- Presence sensor: \$ 41.99
- Single-zone processor: \$75.99
- Conduit box: ~\$19.00
- Other circuit elements: ~\$15.00
- 100 W light bulb: \$1.00
- Entire System: ~\$152.98**

As mentioned in the introduction to the presence sensor, we are trying to solve the issue of efficiently lighting an exterior walkway. In this sense, we expect the presence sensor to pay for itself in just a short amount of time by saving money on a homeowner's electric bill.

We did some brief calculations to find the payback time for this system:

First we assume the average house has their lights on for 10 hours each night. In North Carolina the cost of electricity is \$0.07 per kWh, and we are assuming 100 W bulbs. Thus in one year, for one exterior bulb, it would cost \$25.85.

Now if this house were to use the presence sensor and the same 100 W bulb, we will estimate that a single light bulb would be on for only a total of 10 minutes per night. This amounts to only \$0.42 per year.

By using the presence sensor, we could save about \$25.13 per year per bulb. Since each sensor will light two bulbs, we will save \$50.26 per year. Using these numbers, this system should pay for itself in about 3 years.

There has been talk about using LED bulbs on the exterior of the house. If this were the case, the numbers would change a bit because the bulbs are more expensive but use considerably less power and have a long lifetime. This means the system would probably take longer to pay for itself, but would save us more money in the long run.

We also could save money in the long run by purchasing multi-zone processors as opposed to the single zone processor. This setup, obviously, would allow us to purchase only the one multi-zone processor (costing \$134.99) per every three single-zone processors we would need to buy otherwise. Which and how many processors we buy will be determined once the length of the walkway is known.

Installation Phase & Timeline

In terms of installation, no serious considerations need be taken into account while building the house. We will require that both 120VAC and 12VDC power be sent to each sensor, however this could be installed at a later time. The conduit box will be located beneath particular stepping stones, and thus could simply be dug into place at any time. This project might be labeled as Phase 2. It is not necessary, but it would be advantageous to have them installed prior to move-in so that the front path may be lit for safety and security purposes. Before being installed, however, we recommend that the timing circuit be properly soldered onto a PCI board to make the circuit more robust. Using the breadboard allows for elements of the circuit to more easily be jostled out of place.

Future Advancement

One idea which we did not have the opportunity to pursue is the use of solar-powered walkway lights. Our system would become even more efficient if we could use lights that ran on batteries charged by the sun. This might be advanced even further by powering the processors with solar power. If this was

accomplished, we would have no need to run any lines from the house to the sensors, thus creating a system that would be completely stand-alone.

Conclusion

Energy efficiency and modern technology are two of the more prominent themes of the DELTA Smart House, so to create a system which meets the standards of both is ideal. This is what we have done in the creation of our presence sensor network. By using modern sensing techniques, we are able to make individual lights turn on as a person approaches, and we are able to save energy by having those same lights turn off once that person passes. With this simple project, we are able to capture the essence of two primary goals for the Smart House and also improve the general safety and security of the property.

***See Appendix C for a brief manual for the presence sensor.

Appendix A

Perl script for internet database

```
use DBI;

$DSN = 'MyEasinetEvents';
$dbh = DBI->connect("dbi:ODBC:$DSN", "", "") or die "Could not connect to MyEasinetEvents";
$stmt = $dbh->prepare("SELECT * FROM Events") or die "Could not connect prepare Events";
$stmt->execute() or die "Could not execute SELECT * FROM Events";

$DSN2 = 'MyEasinetEventDescription';
$dbh2 = DBI->connect("dbi:ODBC:$DSN2", "", "") or die "Could not connect to EventDescriptions";
$stmt2 = $dbh2->prepare("SELECT * FROM Events") or die "Could not prepare Event Descriptions";
$stmt2->execute() or die "Could not execute SELECT * FROM Events for descriptions";

@event;

while( @row2 = $stmt2->fetchrow_array )
{
    @event[$row2[1]] = $row2[2];
}

$stmt2 = $dbh2->prepare("SELECT * FROM EventSub") or die "Could not prepare SELECT * FROM
EventSub";
$stmt2->execute() or die "Could not execute SELECT * FROM EventSub";

@eventsub;

while( @row3 = $stmt2->fetchrow_array )
{
    @eventsub[$row3[1]] = $row3[2];
}

print "<html><body><h1><center>Fingerprint Scanner Records</center></h1>";
print "<center><table border = '5' cellspacing = '10' cellpadding = '10'>\n";

print "<tr><th>EventID</th><th>EventTime</th><th>EventType</th><th>EventSubType</th></tr>\n";

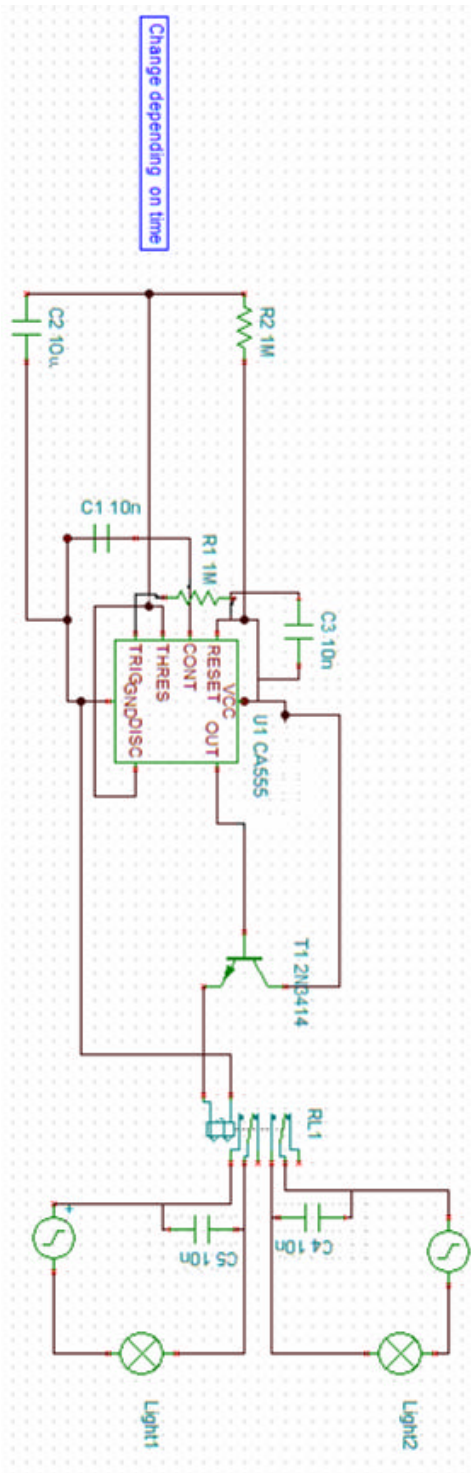
while( @row = $stmt->fetchrow_array )
{
    print
"<tr><td>$row[0]</td><td>$row[1]</td><td>$event[$row[2]]</td><td>$eventsub[$row[3]]</td></tr>\n";
}

print "</table></center>\n";
print "</body></html>";

$stmt->finish;
$dbh->disconnect;
```

Appendix B

Presence sensor schematic



Appendix C

Manual for presence sensor

Introduction

The presence sensor and processor may be purchased directly from SmartHome.com, and, out of the box, are prepared to detect flexing in whatever material the sensor is mounted upon. The sensor works by changing resistance when deformed. The processor recognizes this change in resistance and closes its imbedded relay. Included in the processor is a potentiometer which allows users to determine how sensitive the sensor will be, or, in other words, to what extent flexing in the material should go unnoticed.

In the Smart House, we wish to use these presence sensors to light the front walkway as a person walks along. To achieve this, we need both a DC circuit to run the processor and an AC circuit to run the sidewalk lamps. The best way to include both is to use a DPDT relay. With this device, when the processor sends its signal announcing the tripping of the presence sensor, our relay can close and allow the AC circuit to flow, thus supplying power to light the lamps. Our circuit also requires some timing, for we want lights to stay on for a short period of time after the sensor is tripped. To achieve the necessary timing, we used a 555 timer. These IC chips allow timing based on the values of specific capacitors and resistors included in the circuit. In our case, we want the lamps to illuminate for 10 seconds, thus we use a 10 μ F capacitor and a 1 MO resistor. Now, when the sensor is tripped and a signal is sent to the DPDT relay, the timer allows this relay to remain open for 10 seconds.

Specifications

Input:

Nominal Current Draw: 67 mA
Pull-In Voltage: 12 VDC

Output:

Output Contact Ratings: 5 Amps at 240 VAC
5 Amps at 28 VDC
Output Type: DPDT relay output – normally open
Outputs: 2

Circuit Components

Resistors:

One 1 MO pull-up resistor
One variable resistor/potentiometer (for RC timing circuit)

Capacitors:

One 10 μ F or variable capacitor (for RC timing circuit)
Four 0.01 μ F ceramic

Transistor:

One 2N3414 NPN

IC Chip:

One 555 Timer

Relay:

One DPDT Plug-in relay (i.e. RadioShack: part# 275-206)

Processor:

One Sure Action Inc. presence sensor processor – model 111

Sensor:

One strain gauge presence sensor (i.e. Smarthome.com: part# 790901)

*** For wiring diagram, see Appendix ***

General Hints & Tips

- Ensure correct polarity of can capacitor
- Be sure to place 0.01 μ F capacitor in parallel with relay leads to reduce “bounce” from AC lines
- Be sure to include 0.01 μ F capacitor between positive and negative DC lines to reduce noise



- Transistor leads: **E C B**
- Don't run AC and DC lines within the same encasement – this is against code