Internet of Things (IoT) Home Automation

Submitted by:
Diego Urquiza, Amgad Armanus,
Evan Arbeitman, Felix Mateo, Nirjan Thayaparan

Team Project Number: 3

Advisor:
Yanyong Zhang

April 18, 2015

Submitted in partial fulfillment of the requirements for senior design project

Electrical and Computer Engineering Department
Rutgers University, Piscataway, NJ 08854

Table of CONTENTS

1. Abstract

Smart Home Automation:
Driven by the Internet of Things

The Internet of Things is a fairly new concept that involves the interconnection of devices to make our lives much simpler and more efficient. According to a recent report from Business Insider, "The Internet of Things will be the largest device market in the world. We estimate that by 2019 it will be more than double the size of the smartphone, PC, tablet, connected car, and wearable technology" (John Greenough). Our team is going to capitalize on the Internet of Things market in order to bridge an everyday household with fast growing technology. For instance, have you ever left your house and forgot to turn the lights off? Or, imagine yourself cooking a nice meal and you have an unexpected visitor? Wouldn't it be nice if you can open the door without having to worry about your dinner being ruined? Our group has a plan to create a simple and easy to use Smart Home system that will make mundane tasks a thing of the past.

Our team has created a user friendly mobile application that can control certain features of your smart home such as the front door. We will design an electronic lock that will have two key features; security and remote access. The Lock is powered by an ultra low power MSP430G2 microcontroller also has a small camera module to take a picture of a visitor and send the image to the user's phone. Once the user receives a notification, he or she can choose to open the lock

from their phone. Our team has designed a smart power outlet and a lighting system that can be controlled from the mobile interface. This will allow the user to turn on and off appliances remotely or set schedules where appliances can turn on and off automatically. We have accomplished this by designing and implementing a server that is running on a Raspberry Pi; this server is the central hub that communicates wirelessly to the to our smart lock, smart power outlets, and the lighting system. The user will be able to communicate with the server over the internet using the mobile application. Home automation technologies are quickly becoming more popular, as a result, companies are rolling out new products to enhance the standard home into the smart home. There are a number of companies out there that have already pre-packaged solutions to create a smart home. The problem is that these solutions are too expensive and very limited in scope. They lock the user into a specific set of approved devices that are compatible with their systems. These devices are expensive and sometimes frustrating to use. These issues diminish the value of the return on investment for the user. Our inspiration for our *Internet of Things Home Automation* system is to design a robust application that the consumer can afford.

2. Introduction

The Internet of Things (IoT) is a fast-emerging technology that creates a network of IP-connected devices with the potential to deliver significant business and consumer benefits. Internet of Things is fueled by several factors, including the exponential growth of smart devices, a concurrence of low-cost technologies (sensors, wireless networks, big data and computing power), pervasive connectivity and massive volumes of data. According to a case study, Cognizant[1] claims that by the year 2020, there will be approximately fifty billion devices connected to the Internet. These recent trends have led to several innovations, in particular building and home automation. As the number of IP-connected devices increases, Internet of Things applications can be used to monitor and control mechanical and electrical components such as locks, cameras, light switches, etc. For instance, technologies such as Lockitron, August Lock[2], and Wink Home solutions, are some of the state of the art technologies available on the market that grant consumers the option to upgrade their homes. Our motivation for *Internet of Things Home Automation* is to develop a Smart Home solution system that is more affordable than current market products. Despite the numerous benefits of the Internet of Things, some major concerns include security and privacy. Internet of Things thrives itself on collecting data from the millions of users who are connected to the Internet, as a result, organizations can collect, manage, and mine user data for their own benefit. For instance, smart devices require minimal human interference; as a result, companies need to be aware of hacking and other criminal abuse. Lastly, the technology that drives Internet of Things applications is accessible to just about everyone. Our project is based off the "Do It Yourself" (DIY) principle; consumers have the capability to expand our system to fit their needs.

---

[1] "Reaping the Benefits of the Internet of Things." Www.cognizant.com. Cognizant Reports. Web. 18 Apr. 2015.

[2] http://august.com/

3 Methods / Results / Approach

3.1 Methods

a. System Design and Implementation

i. The *Internet of Things Home Automation* system is focused around a Raspberry Pi single board computer running a Debian Linux Server[3]. In addition, our Linux server is running openHab software that can communicate with other IP-connected devices locally or over the Internet. For our home automation system we addressed three distinct features of any home: a Smart[4] Lock, Smart Outlet, and Smart Surveillance Camera. Compared to other home automation solutions, our system is much more robust and sustainable because of the user isn't limited to the use cases the manufacturers designs. For instance, openHab and Debian are open sourced software packages that are constantly being worked on by other users. This also means that we can easily expand our system by adding Z-Wave or Zigbee devices; simply by adding a Z-Wave or Zigbee transceiver to our Raspberry Pi hub. By utilizing open source packages, our product can be expanded to fit user needs at anytime. In addition, users can manage their Smart devices through either the openHab web client or openHab mobile application.[5]

ii. One of the main features of *Internet of Things Home Automation* is the Smart lock. Smart Lock's mechanical components include a HS-322HD servo motor

[3] Documentation on running Debian software on the Raspberry Pi can be found here http://www.raspbian.org/RaspbianDocumentation

[4] Smart - an IP-connected device that communicates with our openHab software.

[5] Mobile Application is available on iOS and Android devices.

that spins a deadbolt lock in either a clockwise or counterclockwise direction. The servo can be controlled from the ESP8266 WiFi chip that is attached to the microcontroller or it can also be controlled using a momentary push-button that is attached to the microcontroller. Initially, we implemented the Smart Lock using an Arduino Uno microcontroller. However, to produce longer battery life for the Smart Lock, we decided to implement it using the Ultra-Low Power MSP430G2 microcontroller. Using the MSP430G2 we saw a significant reduction in the current draw which lead to longer battery life.

iii. Moving on, another piece of our home automation extends to our Smart Outlet. The main purpose of the outlet was to control power from any wall socket using a microcontroller, in this case an arduino. The Smart Outlet can be broken down into three pieces: the Relay SPST attached to the Relay Control PCB, the GFCI outlet, and finally the arduino. The Relay SPST is simply a large mechanical switch that can be toggled on or off by energizing a coil. The relay will allow us to control the flow of current through the outlet. The GFCI outlet was required for this project because of its ability to detect abnormal amounts of current. When it does, it will assume that amount of current is flowing through your body and could be fatal and will thus automatically shut off. Besides that, the outlet is simply a port to plug an electrical device into. Finally the arduino contains the code that controls the relay board to let current flow in and out. As mentioned before, the ESP8266 WiFi chip is needed to communicate within microcontrollers, in this case between the Smart outlet arduino to the server.

iv. We also created an IP-Connected Smart Surveillance Camera. This camera utilizes a Raspberry Pi and a 5-megapixel camera attachment. The camera is set up to automatically connect to the WiFi network and obtain a static IP address then set up its own web server. This camera provides live surveillance video that can be accessed from the OpenHab web interface or the mobile application. Using open source software Motion on Linux, we are about to monitor the video surveillance and when we see significant change in the image frames, we assume that we have detected motion and automatically capture an image. These images are sent to our main Raspberry Pi server. Using the Dropbox API and a bash script that runs every minute using a Linux Crontab, we automatically sync these images to the user's Dropbox account.

v.

b. Time Constraints

i. Throughout the project life cycle, we experimented with a variety of microcontroller sensors that were not available at common retailers. We realized that creating base models of each piece of our system was within the time constraint, yet we also realized that each part could also be expanded upon and improved as well. For example, with the Smart Outlet we finished the base model in the sense that we were able to successfully control current through the outlet. But if we had more time, we could have implemented a current sensor within the Smart Outlet to track the current usage, so that a user could see where current was being used and how much.

c. Knowledge Base

i. "OpenHAB - Empowering the Smart Home." http://openhab.org
ii. "ESP8266." - NURDspace. https://nurdspace.nl/ESP8266
iii. "Motion for Linux." http://www.lavrsen.dk/foswiki/bin/view/Motion/WebHome

d. Hardware and Software Tools

i. The chart below contains a list of hardware devices and software packages we used for designing our system.

Hardware	Software
1x MSP430 Microcontroller	Raspian - Linux distribution for Raspberry Pi
2x ESP8266 WiFi Chip	openHab Software
2x Raspberry Pi	Fritzing- Circuit Drawing Tool
HS-322HD Servo Motor	Linux Motion
Raspberry Pi 5-Megapixel Camera	Arduino IDE
GFCI Outlet	Energia IDE
Relay SPDT-NO 20A	

e. Existing Standards

i. Standardized network technologies: IEEE 802.11a/b/g, Internet Protocol—IPv4 and TCP.

f. Regulatory Issues

i. One of the key issues we encountered while designing the *Internet of Things Home Automation* system was power management. The Smart Lock uses a Servo motor powered by four AA batteries, and it's controlled by an MSP430 that runs off of two AA batteries. For our system to be practical, we needed to minimize the amount of times the users would need to replace the batteries; to do so, we experimented with both an Arduino and MSP430 to determine which microcontroller was more power efficient. Specifically, we needed to determine which microcontroller drew the least amount of current during the idle state. As a result, we derived this equation and determined that the MSP430 had much longer battery life while in the idle state.

Estimated Battery Life

The Servo motor uses four AA batteries with an average of 2200 mAh

Average Current Draw for the Servo Motor

Duty cycle = T(on)/(T(on)+T(off))*100

Assume that servo is on for 5 seconds each hour

T_{on} = 0.083 minute

T_{off} = 59.917 minutes

Duty cycle = 0.083/(59.917+0.083)*100 = 0.138%

Average current draw per hour = 0.00138 * 160 mA = 0.22 mA

2200 mAh /0.22 mA = 10000 hours

10000 hours / 24 hour = 416 days

Battery Life comparison between Arduino Uno and the Ultra-Low Power MSP430 implementation of the Smart Door Lock

	Arduino Uno	**MSP430G2**
Average current draw	50 mA	400 uA
Average current draw of ESP8266 WiFi chip when on stand by	0.9 mA	0.9 mA
Estimated Battery Life (hours)	2200 mAh / 50.9 mA = 43.22 hours	2200 mAh / 1.3 mA = 1692.3 hours
Estimated Battery Life (days)	**43.22 hours / 24 hours = 1.8 days**	**1692.3 hours/24 hours= 70.5 days**

Note: The Arduino Uno uses four AA batteries as its voltage input. However, the MSP430G2 only uses two AA batteries.

3.2 Use of Standards

Some examples of standards that might impact design choices:

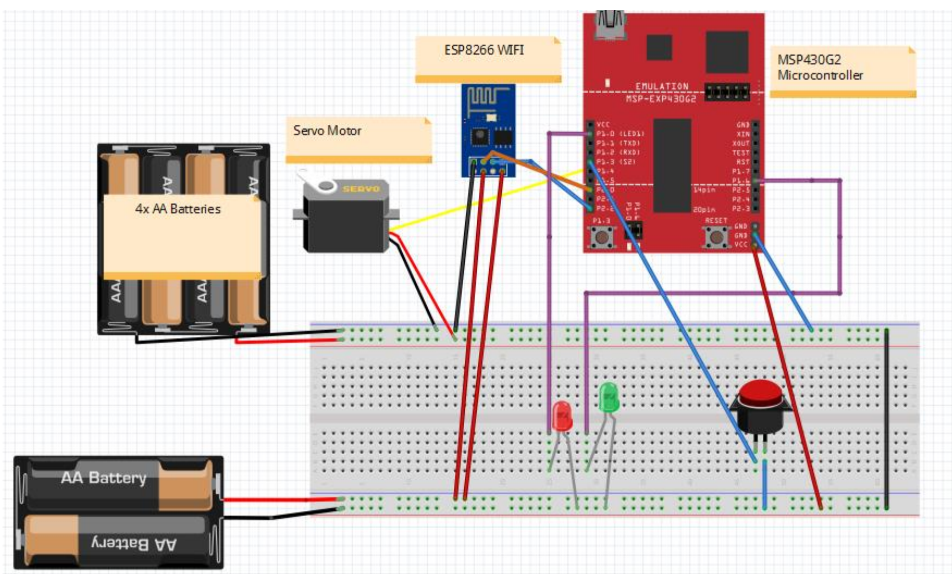
a. Standardized network technologies:IEEE 802.11a/b/g, Internet Protocol—IPv4, TCP.

b. Standardized software development tools, and software environments: Arduino IDE, Energia IDE, Esplorer Lua IDE.

c. Open source standards, software, and operating systems: Linux, Apache server, OpenHab Server.

3.3 Experiment / Product Results

Smart Lock (High Level Schematic)

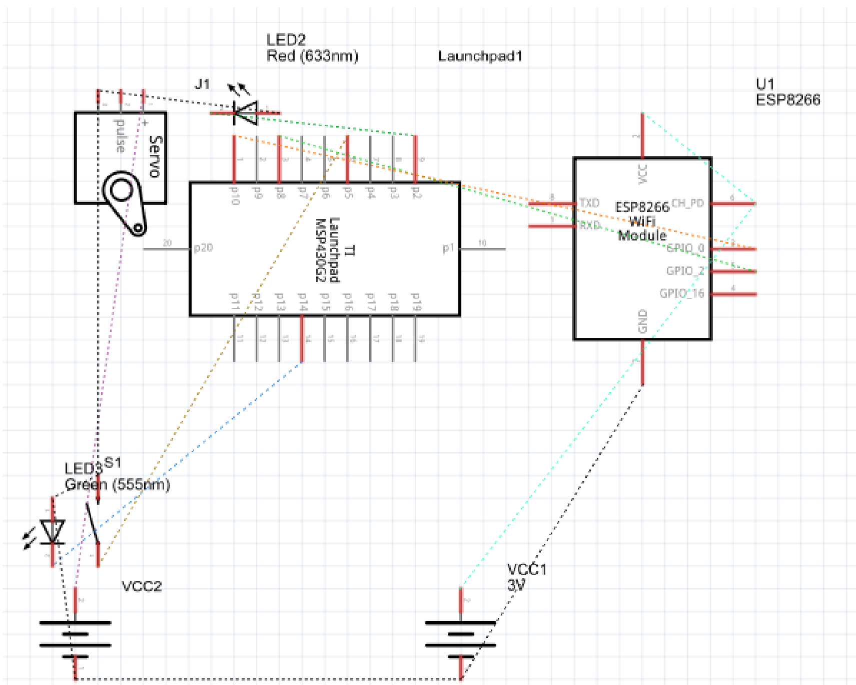


Implementation

At the heart of our Smart Lock is an MSP430 microcontroller. Along with a WiFi adapter, the microcontroller communicates with our Raspberry Pi web server.The web server will send a signal to the microcontroller and execute C code that will control the servo motor. The C code will spin the lock either 180 degrees counterclockwise or clockwise; similar to the motion of manually unlocking a door.

The Smart Lock was powered by a total of 6 AA batteries: 4 to power the microcontroller and 2 to power the servo. Initially, we tried powering the entire lock with 4 batteries however, that wasn't sufficient. In our early experiments, we observed that the servo motor wasn't being supplied with another power, resulting in fewer cycles of the servo motor.

Smart Lock (Low Level Schematic):

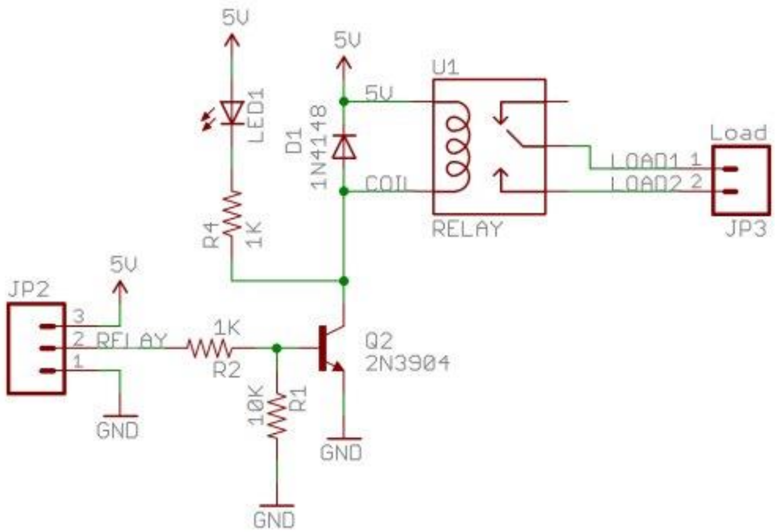


Sample Code for Smart Lock Implementation for the microcontroller

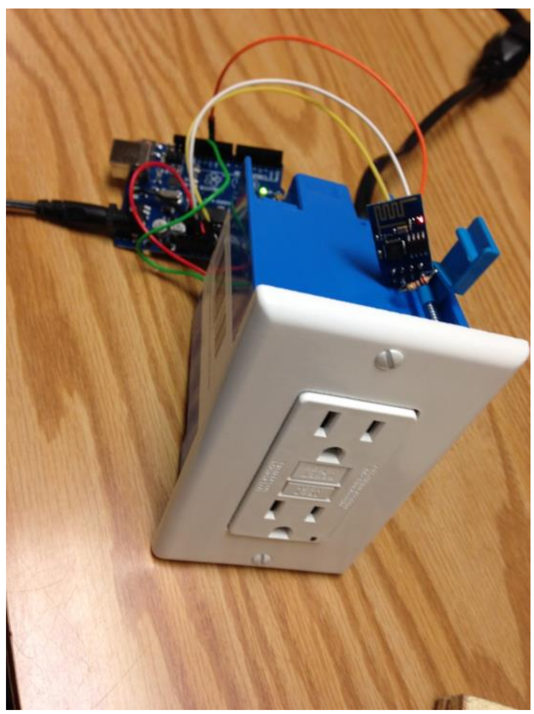
```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
// twelve servo objects can be created on most boards
int pos = 10;    // variable to store the servo position
int buttonInput = 7;
int GPIO_0 = 12;
int GPIO_2 = 13;
int locked = 1;
int greenLedOut = 6;
int redLedOut = 5;
// Button debounce
int pinState = HIGH;         // the current state of the output pin
int buttonState;             // the current reading from the input pin
int lastButtonState = LOW;   // the previous reading from the input pin
// the following variables are long's because the time, measured in miliseconds,
// will quickly become a bigger number than can be stored in an int.
long lastDebounceTime = 0;  // the last time the output pin was toggled
long debounceDelay = 100;    // the debounce time; increase if the output flickers
void setup()
{
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
  pinMode(buttonInput, INPUT_PULLUP);
  pinMode(GPIO_0, INPUT_PULLUP);
  pinMode(GPIO_2, INPUT_PULLUP);
  pinMode(greenLedOut, OUTPUT);
  pinMode(redLedOut, OUTPUT);
  Serial.begin(9600);
  myservo.write(10);
  delay(5);
  digitalWrite(redLedOut, LOW);
  digitalWrite(greenLedOut, HIGH);
  Serial.println(pos);
}
void loop(){
  Serial.println(pos);
  int reading = digitalRead(buttonInput);
  if (reading != lastButtonState) {
    // reset the debouncing timer
    lastDebounceTime = millis();
  }
  if (digitalRead(GPIO_0) == HIGH && digitalRead(GPIO_2) == HIGH && locked == 0) {
    myservo.attach(9);
    //LOCK DOOR
    for (pos = 121; pos >= 1; pos -= 1) // goes from 180 degrees to 0 degrees
    {
      myservo.write(pos);              // tell servo to go to position in variable
'pos'
      delay(5);                       // waits 15ms for the servo to reach the
position
    }
    locked = 1;
    digitalWrite(redLedOut, LOW);
    digitalWrite(greenLedOut, HIGH);
```

```
    myservo.detach();
  }
  else if (digitalRead(GPIO_0) == LOW && digitalRead(GPIO_2) == LOW && locked == 1) {
    myservo.attach(9);
    //UNLOCK DOOR
    for (pos = 0; pos <= 120; pos += 1) // goes from 0 degrees to 90 degrees
    { // in steps of 1 degree
      myservo.write(pos);              // tell servo to go to position in variable
'pos'
      delay(5);                       // waits 15ms for the servo to reach the
position
    }
    locked = 0;
    digitalWrite(redLedOut, HIGH);
    digitalWrite(greenLedOut, LOW);
    myservo.detach();
  }
  else if ((millis() - lastDebounceTime) > debounceDelay) {
    if (reading != buttonState) {
      buttonState = reading;
      // only toggle the LED if the new button state is LOW
      if (buttonState == LOW && locked == 0) {
        myservo.attach(9);
        //LOCK DOOR
        for (pos = 121; pos >= 1; pos -= 1) //goes from 180 degrees to 0 degrees
        {
          myservo.write(pos);  // tell servo to go to position in variable 'pos'
          delay(5);           // waits 15ms for the servo to reach the position
        }
        locked = 1;
        digitalWrite(redLedOut, LOW);
        digitalWrite(greenLedOut, HIGH);
        myservo.detach();
        // ledState = !ledState;
      }
      else if (buttonState == LOW  && locked == 1) {
        myservo.attach(9);
        //UNLOCK DOOR
        for (pos = 0; pos <= 120; pos += 1) // goes from 0 degrees to 90 degrees
        { // in steps of 1 degree
          myservo.write(pos);  // tell servo to go to position in variable 'pos'
          delay(5);    // waits 15ms for the servo to reach the position
        }
        locked = 0;
        digitalWrite(redLedOut, HIGH);
        digitalWrite(greenLedOut, LOW);
        myservo.detach();
      }
    }
  }
  else {
  }
  lastButtonState = reading;}
```

Smart Outlet Circuit



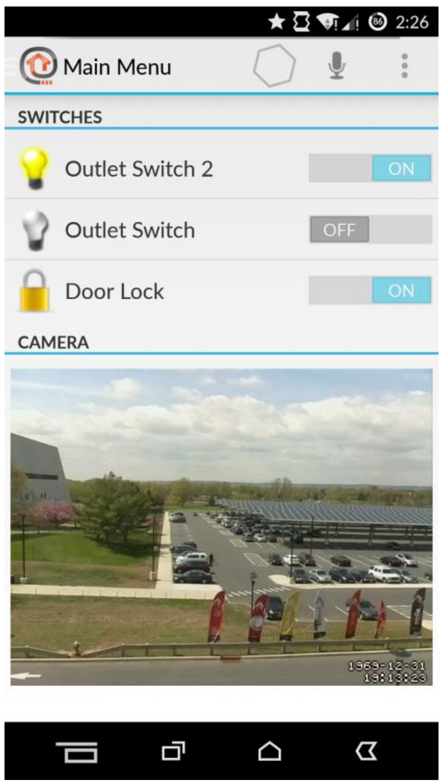
Smart Outlet



Smart Outlet Implementation:

The above circuit sums up the internal workings of the Smart Outlet. As one could see, the Relay SPDT contains the coil on the left side, which is required to toggle on or off the current. The coil requires up to 80 mA to operate, which is more than a GPIO pin can handle. Thus the circuit requires a transistor, specifically an NPN transistor which can handle up to 200mA, as a controllable connection to ground. When the CTRL pin goes high, the transistor connects to ground, sending current through the coil. In the case that anything goes wrong with this operation, we connected R1 to the RELAY pin. The above circuit also contains a 1N4148 diode in the case that when the coil de-activates and tries to suppress current charge. The diode will forward bias causing the current to flow back to the 5V rail.

Mobile Application User Interface



Once the user opens the mobile application, they will see all their connected devices and their status. They will also be able to view a live feed from the surveillance camera. User is also to interact with their devices from this screen.

Product Comparisons:

	SmartThings Hub	Our Smart Hub
Operating System	RTOS - Can only be modified by manufacturer	Raspbian - Linux Distribution - Open Source
Device Connectivity	IP, Z-Wave, Zigbee	IP, Z-Wave can be easily added using a Z-Wave USB stick.
Range	100 to 150 feet	100 to 200 feet - Can be easily expanded by using a USB WiFi antenna.
Internet Connectivity	Ethernet	Ethernet, WiFi
Price	$100	$45

	August Lock	Our Smart Lock
Price	$249	$35
Keyless	Yes	No. In the event that the user doesn't have access to their phone, they can still enter open the lock.
Presence Detection	Yes	Yes
Internet Connectivity	WiFi	WiFi
Bluetooth	Yes	No. A bluetooth module can easily be added at anytime.

	Nest DropCam	**Our IP-Surveillance Camera**
Operating System	Unknown	Raspbian - Linux Distribution - Open Source
Connectivity	WiFi	WiFi
Storage	Cloud storage at Nest - Requires yearly subscription	Local storage and Cloud Storage at Dropbox
Motion Detection	Yes	Yes
Live Video Stream	Yes	Yes
Price	$199 + $299 per year for 30 day cloud recording.	$70

	Belkin Wemo Switch	**Our Smart Outlet**
Microcontroller	Unknown	Arduino Uno
Connectivity	WiFi	WiFi
Number of outlets	1	2
Price	$50	$30

3.4 Cost and Sustainability Analysis

Sustainability consideration and constraints includes economic, environmental, and social (equity) aspects that need to be evaluated and taken into account in project research and development. There is a strong relationship between these three pillars of sustainability. These need to be considered and incorporated in this section with a discussion on their design constraint and the positive and negative effects of the project within this scope.

Beyond the feasibility of the technical solution an engineering project needs to take into account the following aspects:

a. Economics (cost) impact:

   i. Our *Internet of Things Home Automation* system benefits the consumer in two ways:

      a. Inexpensive Solution for Home Automation

      b. Opportunity to Expand Product

   ii. Some of the Smart Home systems available on the market include: Wink[6], Athom Homey[7], August Lock[8], etc. have a manufacturer suggested retail price of $300, $243, and $199 dollars, respectively. Unlike these products, *Internet of Things Home Automation* bundles lock automation, home security, and power management system into an all-in-one system for an affordable price.

   iii. A very different sort of economics comes into play when connected devices begin to communicate with other devices. Through the network effect[9], as the number of users increase, the value of smart devices also increases. Unlike other smart home systems,

---

[6] http://www.wink.com/products/wink-relay-touchscreen-controller/
[7] http://www.cnet.com/products/athom-homey/
[8] http://august.com/
[9] http://www.technologyreview.com/news/527361/the-economics-of-the-internet-of-things/

Internet of Things Home Automation is not limited by manufacturer constrictions. For instance, consumers who purchase our system can expand it to their needs, whereas other systems are not expandable.

b. Social impact of the product: when relevant, please consider

i. Our *Internet of Things Home Automation* system is designed for the sole purpose of making people's lives easier. With remote access to any network connected devices, consumers can monitor their homes from a mobile phone with only a few clicks of a button. For instance, the Smart Surveillance camera can detect changes in motion and alert the user when someone appears at the front of their door. At that point, the user can identify who is outside their door, and then decide to unlock the door from their phone.

ii. One tradeoff that exists with Internet of Things technologies is data security and privacy. As the number of devices connected to the Internet increases, consumer data becomes more vulnerable to data mining and cyber attacks.

iii. *Internet of Things Home Automation* is the ideal product for personal use. Home automation is a booming trend, as more and more people are looking to upgrade their homes into Smart Homes. For example, with the Smart Outlet Current sensor, consumers can keep track of power consumption to reduce their energy bill each month.In addition, the Smart Surveillance camera periodically updates the user when someone appears at their front door.

iv. With our smart outlet we plan to keep tract of power consumption rate that the user can see how much money he can save by either changing to a more energy efficient device or changing their usage patterns.

v. This product does automate the process of getting into your home but does not remove the key feature.

vi. This product creates a lot more jobs in the internet of things market. We are adding an alternative to come into your house without removing the physical key feature.

vii. Our product aims at securing any door to your house by receiving notifications and a log folder of everyone who walks across one of our security cameras.

4. Conclusions / Summary

Technology is always advancing and the Internet of Things may soon be the future of home automation. It's aim to connect a multitude of different devices and people together in one giant network, the Internet of Things inspired our smart lock, smart outlet, and smart lighting systems. Although these technologies are already in existence, they can be troublesome to use and expensive to own. Our systems was created with affordability and efficiency in mind. Also, because our systems were designed out of open source frameworks; end users have the ability to alter our creations and make it their own.

Throughout the project lifecycle, one of the biggest challenges we encountered was optimizing power consumption. During the early phases of testing, we observed that our smart lock was drawing too much current and as a result, the battery life was significantly shorter. To solve this problem, we experimented with the MSP430 microcontroller and compared our results with the Arduino. We determined that the MSP430 drew much less current in the idle state, therefore prolonging the battery life.

Acknowledgments

Special thanks to Professor Yanyong Zhang and Professor Bernhard Firner for their support throughout the project.

References

[1] "Quick Links." *RaspbianDocumentation*. N.p., n.d. Web. 20 Apr. 2015.

[2] Regalado, Antonio. "The Economics of the Internet of Things." 20 May 2014. Web. 20 Apr. 2015.

[3] "Home Automation for Geeks." Home Automation For Geeks. Web. 20 Apr. 2015.

[4] "Arduino Controlled Relay Box." Instructables.com. Web. 20 Apr. 2015.

[5] "WebHome." WebHome. Web. 20 Apr. 2015.