**Passive Infrared Sensor Doorbell**

Seth Carpenter, Austin Cheng

Illinois Institute of Technology

**Introduction**

This project was undertaken in order to develop a prototype which would utilize embedded systems and demonstrate elements of data collection and data transmission. Our inspiration came from the consideration that those suffering from hearing loss of varying degrees do not have a full home solution for notifications that one is at their door. The traditional doorbell is an age old method for notifying a resident that there is someone calling for their attention. While this has been an effective method for the most part, we saw potential to integrate the technology of today into a system that could provide a more advanced notification system. This system would also be more effective for use for those with hearing loss than existing solutions. It also provides a base for further development with the potential to be built upon to increase its functionality and add features.

**What does it do?**

This project senses when someone comes up to the door you have the sensor placed at. When a presence is sensed, a signal is sent indoors to a listening and controlling component that flashes your Philips Hue lights a couple times to alert you of the situation. You can then press a button from the inside that flashes another Hue light located by the door to signal that you have received the notification and are coming to the door.

**Modules**

The key modules of this system are two XBees, two Arduino processing units and the Philips Hue lighting system. The XBee is a quarter-sized wireless networking module. They can be configured to exchange data using the Zigbee protocol. It is possible to link a network of these modules together in order to create a mesh network which is coordinated by a single XBee. They can either collect and transmit data on their own using Zigbee protocol data units or be configured to send data in a serial format from one processing device to another. Our project utilizes the latter configuration with the XBees communicating in the RT format. The Arduino is an inexpensive microcontroller which is programmed through a multi-platform compatible IDE. In this project, it is used as both a control element as well as a power source for the XBees and other elements connected to it. The Hue lighting system is an advance home lighting solution developed by the electronics manufacturing company Philips. This lighting system utilizes bulbs with the ability to display a wide range of colors which are primarily controlled by smartphone or computer applications. The system includes a network enabled bridge which communicates wirelessly with the lights using the Zigbee protocol. The bridge acts as a local server on the network and as such can be queried using HTTP requests to receive and send JavaScript Object Notation formatted commands from any device on the same network. Development of control applications is open source, which is how we were able to integrate this system into our project.

**Outdoor Component**

The door component consists of an Adafruit passive infrared sensor, an Arduino, and an XBee. The Arduino is powered by a USB cable that is plugged into a standard wall socket adapter. The PIR is then powered by the 5V output of the Arduino while the XBee is powered by the 3V output of the Arduino. The XBee is configured as an AT router and its transmit and receive pins are connected to the respective pins on the Arduino. The router is used to exchange serial communications between the two Arduinos in the system. The PIR sensor has a single analog output which is connected to the analog 0 pin on the Arduino. The sensor outputs a cone shaped field of infrared light. As long as there is no change in the field, the sensor sends its voltage out of its ground connection. When a change is detected, the sensor sends the voltage through its output pin to the Arduino. The Arduino reads this voltage as an analog value. The Arduino is programmed to respond when the value is above a threshold. This threshold was set just below the average value obtained from testing to compensate for variation. The Arduino responds to the sensor, by sending an ASCII character through the XBee to the component inside the house. Like the indoor component, the outdoor component is a standalone system. Additional sensors can be added and the Arduino can be programmed to both accommodate new sensors as well as receive commands from the other end to trigger additional attachments such as LED’s or motors. The system is limited by the voltage and amperage output of the Arduino. If too many additional components are added in series with the existing components, the voltage will drop and the XBee may not get enough power to function correctly. This may be somewhat mitigated by reversing the power sources from the Arduino and calculating the voltage draw resulting in powering multiple components in series so that the XBee continues to receive exactly 3.3V.

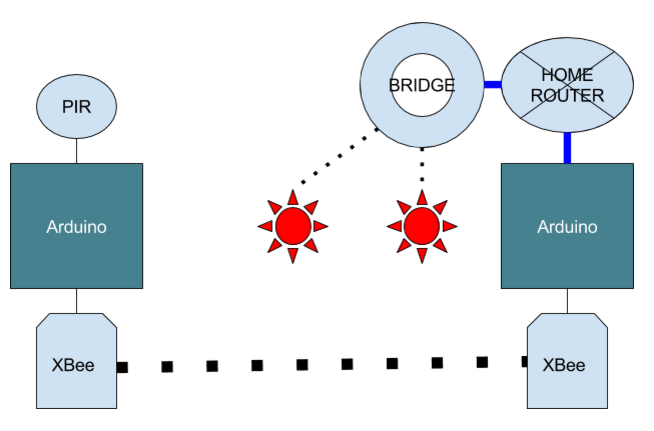
**Indoor Component**

The indoor component of our system is comprised of a second Arduino, ethernet shield, XBee, and the Phillips Hue Lighting System made up of a bridge and the actual LED lights that communicate with the bridge. An existing home network is also needed. As mentioned, an ASCII character is sent by the outdoor component. This character is received by the XBee on the indoor side and sent to the Arduino. The Arduino takes this as a call to send an alert to the lights. A web client runs on the Arduino. This client sends HTTP PUT requests containing JSON over the network  to the Hue bridge which in turn changes the state of the lights specified in the JSON. Our code runs these commands twice with a delay in between to achieve the desired flashing effect. To insure that your current light configuration is not overwritten by the alert sent when the outdoor sensor is triggered, the current state of the bulbs is stored in variables before the alert is sent. It does this by sending HTTP GET requests through the home network to the Hue bridge. The JavaScript Object Notation formatted response is then parsed in the Arduino to store the values of a light relating to its state (on/off), its color, the saturation of the color, and its brightness (Philip Hue API).

There was an error in the example code we were referencing (Bruce). We discovered that the findUntil function required to parse the incoming JSON only takes a character as the second parameter. ("Arduino - Streamfinduntil"). This meant it had to be in single quotes, not double as we were shown. It took us a while to figure out why we couldn't receive the right data when querying the current state of the lights.

**Project Elements**

The data collection of this project comes from the outdoor component primarily. This is achieved through the passive infrared sensor. It is a simplistic example of data collection as the sensor only effectively has a high and low state. Additional components like a camera would be a more complex example of data collection and would require not only much more adaptation of the system, but possibly more computing power than is currently available with the Arduino. The secondary data collection aspect of this system is in the ability of the network enabled Arduino in the indoor component to retrieve and temporarily store the state of the Hue lights. This project ran into some complications regarding electricity. When attempting to create the outdoor component, we initially omitted the Arduino. In that scenario, we sought to use only the XBee and the infrared sensor powered in serial by the USB wall adapter connected to a converter board used to charge batteries at 3.7 V. This proved to be insufficient power for the XBee so we attempted to power each individually with two adapters. While the sensor was able to function in both cases, the XBee was incompatible with the 3.7 V power source and that idea was scrapped. In the end, we decided to use the Arduino in order to provide power as well as add potential for additional functionality. The indoor component had fewer iterations as it already included an Arduino. During this process, the indoor component went from being a coordinator in API mode to a coordinator in RT mode in order for the Arduinos to communicate. The system did not include any additional electrical components as the power was provided directly from the Arduinos and was not manipulated with any resistors or transistor. A button was added to the indoor component in the final iteration of the project. This button was connected to the voltage rail of a breadboard and the other end lead to a digital pin on the Arduino. When pushed, the pin read a digital high and the Arduino sent a command to the Hue bridge to cause a Hue light positioned by the door to blink green in acknowledgement. The data transmission aspect of this project lies in the communication between the XBees as well as the communication between the indoor Arduino and the Hue bridge. The Arduinos are able to send single ASCII characters between each other and the programming within each defines the response to the character received. The Arduino with the Ethernet shield is connected to the home network to which the Hue bridge is also connected. In our system, the light’s states are collected individually before the response commands, both the sensor response and the button response, are sent. After those responses run, the Arduino uses the stored state data to send a command to return the lights to their previous states. Figure 1 shows the overall infrastructure of the project.

Figure 1.

**Cost and Competitors**

The way our prototype is currently built does not compete in price with existing light-based doorbells. To get the price down, a stripped down version of the Arduino would be used. There are doorbells with lights on them that range in price from $10 to around $200, whereas our prototype would run around $311 if you don't already own the Philips Hue Lighting System.  An advantage of our product is how it integrates with an existing system and is much more customizable. The user doesn't need to have another light just for the doorbell. Also, a Hue Light is easier to see than some of the lights on competing lighted doorbells. Our project also includes the ability to acknowledge the person at the door with another light. Obviously, for someone who already owns the Phillips Hue lighting system, this is a more appealing product. Table 1 shows a breakdown of the costs of our prototype. These prices come from the Adafruit website (adafruit.com ) as well as amazon.com.

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Cost** | **Quantity** | **Total** |
| Arduino | $25 | 2 | $50 |
| Ethernet Shield | $45 | 1 | $45 |
| XBee | $23 | 2 | $46 |
| XBee USB Adapter | $25 | 2 | $25 |
| Hue Lighting System | $125 | 1 | $125 |
| Misc Wires/Breadboards | $20 | 1 | $20 |
|  |  | **TOTAL** | **$311** |

Table 1.

There are a variety of alternatives that provide a flashing light in addition to a chime sound. Our motion sensing is another feature over many of these products. Most of these require the person at the door to press a button and have chime sounds built in. Here in Table 2, a few competitors are shown along with their price. These products were found on amazon.com and harriscomm.com.

|  |  |  |
| --- | --- | --- |
| **Product** | **Cost** | **Source** |
| Heath Zenith SL-6144-A | $53.98 | amazon.com |
| Bedrocker BCF-1T1R-B | $18.59 | amazon.com |
| Bellman & Symfon Care Home Alerting Solution | $198.00 | harriscomm.com |

Table 2.

Additionally, many of these products are comprised of merely two components, one for the doorbell, and one as the indoor notification component. With our system, the Hue lights can be placed in multiple locations in a residence in any number and as such can provide its functionality in any configuration or location. This system is not limited to use by only the hearing impaired, however. This system can be useful in cases where a doorbell might not be heard by those with regular hearing due to other circumstances. For example, if the residents outdoors and out of range of a traditional doorbell, a light pointing at their backyard could notify them of additional guests arriving, or in a situation where residents are in their basement or bedroom watching a loud movie, or using their computer with headphones on, this system can be beneficial. All in all, this system already has far greater functionality than a regular doorbell.

**Future Potential**

There is a lot of flexibility provided by the components in this project. This allows for a wide range of new and exciting features. In addition to using lights to indicate someone at the door, vibrations could also be used although the placement of the vibration device would be limited to the area near the indoor Arduino.  There are products out there that already do this, but they do not integrate into devices you already own, such as your smartphone. A camera would be a useful addition to this setup. It would require more powerful circuit boards to process all the data (perhaps an Arduino Mega or an Intel Galileo), thus driving up the cost, but this could also be integrated into your smartphone, sending you text alerts with a picture of whoever is at the door. Another potential feature would be for the integration of the doorbell circuit with a pin on the Arduino, causing the lights to also flash whenever the doorbell is pressed, giving the system the ability to express urgency depending on the frequency of the doorbell presses. This system also shows the potential for applications beyond residential technology integration. Provided the system has a sufficient power source, it can be used in outdoor settings such as campgrounds or remote military camps or bases where detection of intruders may be beneficial. Instead of using the Hue lights as the notification element, simple LEDs and buzzers/speakers can be used to provide the notification.

**Conclusion**

This system is a step towards full smart home technology integration. The system itself already integrates with an advance home lighting solution and has the potential for even more than that. While the price of the system in its current state is relatively high, further optimization can make for a very enticing product for the tech-savvy. This is to say that it is not limited to the original purpose inspired by the idea of helping the hard-of-hearing. This project was an exploration of the very same goal that the technology giant Microsoft is working towards, which is the addition of technology into homes everywhere. The purpose of which this serves is to enhance the lives of residents using integrated systems and as technology develops, we are sure to see this trend grow. Soon, homes could be looking very different from how they look now. The depiction of futuristic smart homes that has been, up until now, a mere figment of the imagination, is now closer to reality than ever.

**References**

"Arduino - Streamfinduntil". *Arduino.cc*. N.p., 2016. Web. 3 May 2016.

Bruce, James. "How To Control Philips Hue Lights From An Arduino (And Add A Motion Sensor)". *MakeUseOf*. N.p., 2015. Web. 3 May 2016.

"Philips Hue API | Philips Hue API". *Developers.meethue.com*. N.p., 2016. Web. 3 May 2016.

Special thanks to Emily Darnell for the idea of using the Philips Hue as a silent doorbell.