HoloLens Internet-of-Things Smart Room

Saluki Engineering Co.

DESIGN REPORT

Team: S17-71-HOL1

Client: Mr. Howard Lo

December 5th, 2017

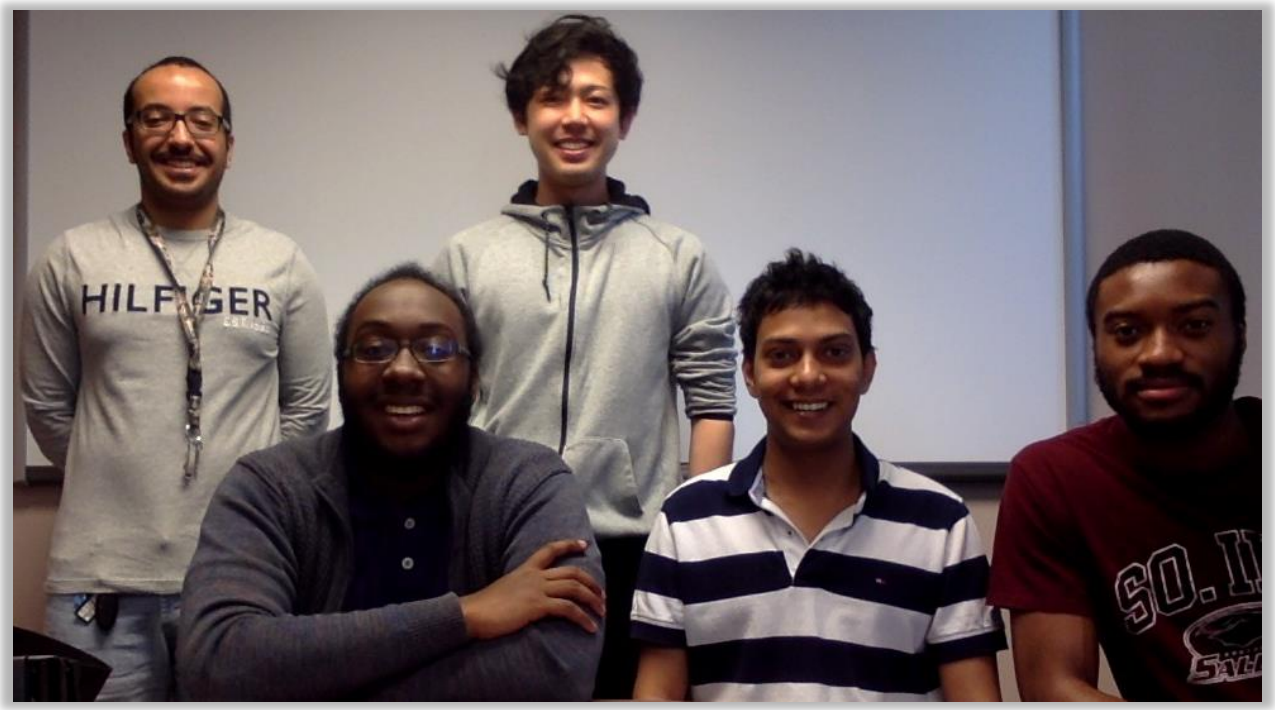
Version 1.0

Team Organization

Shivam Kundan	Computer Engineering	shivamkundan@siu.edu
Isaac Onoh	Electrical & Computer Eng.	isaaczikky@siu.edu
Alaa Alotaibi	Computer Engineering	alaa@siu.edu
Ken Nakazawa	Computer Engineering	ken.nakazawa@siu.edu
Antonio Pugh	Computer Engineering	apugh19@siu.edu

Faculty Technical Advisor

Dr. Iraklis Anagnostopoulos
Professor of ECE, SIU Carbondale
Email: iraklis.anagno@siu.edu



Alaa, Antonio, Ken, Shivam, Isaac

Transmittal Letter [KN]

12/01/2017

Saluki Engineering Company
1263 Lincoln Drive
Carbondale, IL 62901

Mr. Howard Lo
howardlo@outlook.com
Atlanta, GA

Dear Mr. Howard:

On behalf of Saluki Engineering Company, I am pleased to present to you our design report for a HoloLens IoT Smart Room. I am certain you will find the information in line with you needs. The design report will cover the points of key we discussed and the frame of this project:

The system is designed to provide and show creative augmented environment; We created an IoT Environment from scratch for HoloLens. We are able to walk around and interact with holographic and real objects. With interacting by gesture of users, each object will show the value by interacting.

Saluki Engineering Company is a Full-service programming & manufacturing company dedicated to qualifying products and superior customer service.
Thank you for the opportunity to serve you. If you have any question concern or comment, please contact us by following information. We look forward to meeting with you again for future projects.

Best Regards,

Shivam Kundan
Project Manager
Saluki Engineering Company
(217) 974-5324
shivamkundan@siu.edu

Table of Contents

1. Executive Summary [SK]

With almost 15 billion [1] internet-connected devices in use today, we are already living among a deluge of raw information. But all too often, we miss out on the most crucial of insights because this data is not optimally processed or presented. Augmented Reality is all about providing people with real-time data in a way that allows them to do much more with it, essentially exposing intelligence in a manner that is highly consumable.

The HoloLens IoT Smart Room not only bridges the gap between real-time data collection and consumption, but also enables the control of previously “non-smart” analog appliances such as fans, lights, and virtually anything with an on/off switch. By combining the two features, several ‘**Smart Services’** can be implemented. For example, if the temperature in the room is detected to be above a certain amount, we can turn on the fan automatically. The night lights turn on and off automatically depending upon the ambient light level. If the room is supposed to be locked but someone enters, a message can be sent or an alert can be played through the speakers.

Analyzing the heaps of collected data leads to the discovery of new and unexpected Smart Services. During testing of HoloLens IoT Smart Room system, the analysis of temperature, humidity, and brightness data available to us helped to determine the patterns of room occupancy. By studying almost 12,000 points of data collected over 4 days, we were able to tell when the room was occupied, roughly how many people were present, what section of the room was likely occupied, and even the time of twilight in the morning. (Appendix H)

Using this data, the system can potentially co-ordinate with the thermostat to optimize energy usage, yielding significant cost savings, or implement a security system which keeps the occupant’s identities anonymous, while also keeping effective tabs on the room and its contents.

In the next iteration of our product, SEC aims to make the Smart Room predict the needs of occupants based on past observations. For example, it can determine the average time of arrival for people on each day of the week and then heat/cool the room before they arrive, or begin to brew a cup of coffee when the GPS signal on their phone indicates they are about to reach the workplace. Using helpful API’s for web services, it is possible to cross-reference sensor data with information like live weather, current traffic, and bus schedules, to name only a few.

Using HoloLens’s advanced AR capabilities, we were able to bring the Smart Room to life. It’s remarkable responsiveness and versatility allowed for the creation of a holographic user interface that is both beautiful and extremely intuitive at the same time.

[Video: youtu.be/ K9n25WHqLxM]

2. ABET Requirements [SK]

Combining two distinct but rapidly merging technologies into one project required skills and learning that were broad in scope. In addition to building upon knowledge from almost all previous engineering courses, the team members also acquired many new skills over the course of the project.

Some of the classes that we drew upon were –

1. **ECE 296: Software Tools for Engineers**
   Skills: Python, using Raspberry Pi, Arduino, sensors, NXT robot

2. **ECE 327: Digital Circuit Design**
   Skills: Wiring digital devices on breadboards, Combinational & Sequential Logic

3. **ECE 329: Computer Organization and Design**
   Skills: General knowledge of computer architecture

4. **ECE 412: Ad-Hoc Mobile Networks**
   Skills: OSI Network Model, TCP/IP protocol, general knowledge of networks

Outcome →	a	b	c	d	e	f	g	h	i	j	k
Assessed →	x		x		x		x		x	x	x

Fig 1: ABET criteria assessment for ECE 412

5. **ECE 432: Programming for Parallel Processors**
   Skills: Multithreading with pthreads, scheduling

Outcome →	a	b	c	d	e	f	g	h	i	j	k
Assessed →	x		x		x				x		x

Fig 2: ABET criteria assessment for ECE 432

6. **ECE 493: Systems Programming**
   Skills: OS scheduling, Inter-process communication

Newly acquired skills –

1. C# and .NET programming
2. Use of TCP/IP protocol.
3. Two-way communication b/w server and multiple clients.
4. I^2C Communication: Digital sensor boards and LCD display
5. Using IoT Electrical relays
6. Presentation, patience, perseverance

3. Product Function [SK]

The HoloLens-IoT Smart Room has two simple functions –

1. Connect physical devices and sensors to internet.
2. Interact with devices and sensor data in an intuitive way.

4. Design Description [SK] [IO]

4.1 Introduction [SK]

At the topmost level, the HoloLens IoT Smart Room consists of two independent subsystems that communicate with each other using messages sent through the internet.

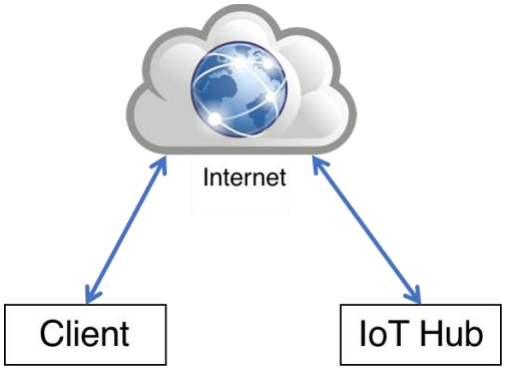


Fig 3: Overview

Client:
Device: Microsoft HoloLens
Purpose:

- Use AR to interact with the room
- Read sensor information
- Additional holographic features

IoT Hub:
Device: Raspberry Pi 3 Model B
Purpose:

- Accept incoming messages
- Send status and sensor information
- Control physical devices

The IoT hub is capable of running independently and monitors the room continuously, even when no clients are connected. This feature is useful for two reasons –

1. Expanding the client to any internet-enabled device such as a smart phone, computer, or hardcoded remote control.
2. Running the room's automated control and monitoring functions (Smart Services).

4.2 Constraints [SK]

4.2.1 Design Constraints

The only design constraint for the product was to use a HoloLens as one of the IoT clients.

4.2.2 Development Constraints

- Budget: $400
- OS: Windows 10 Education/Pro

4.3 Client Side Architecture [IO]

4.3.1 Vuforia

As a team, we decided to add detection capability to our project, and for that we needed to have the ability to identify an object as well as possibly track the object. To help with the implementation of this, we used the software Vuforia, for detection as well as tracking the object.
Initially, we planned to track a three-dimensional object, but we soon found out that for this to be done efficiently, we needed a three-dimensional object with unique data points, as well as a higher resolution scanner to get in detail our unique data points. We decided to switch from three-dimensional object detection to two-dimensional image detection.

Technical Details

Fig 4: Image Target

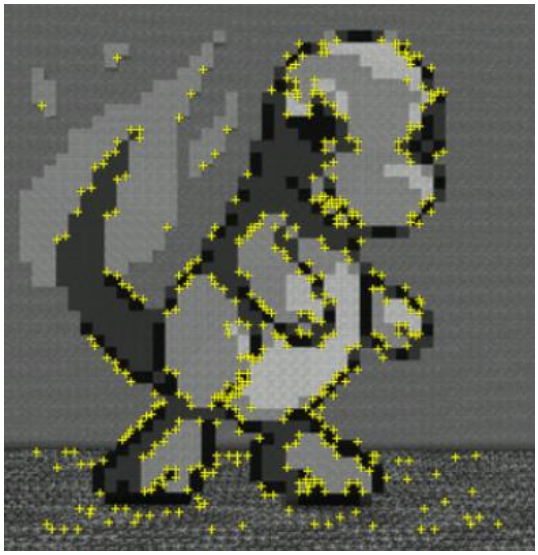


Fig 5: Image Target with Data Points

Using the image target, Vuforia allows you to create a database for the image that's to be detected, and based on this, multiple objects can be bound to a particular image. Also, you can have different image targets being tracked at the same time.
Extended Tracking is Vuforia's counterpart to spatial mapping on the HoloLens. This box needs to be checked when building Vuforia applications for the HoloLens. This way the Vuforia application maps out the environment, and is able to have objects anchored to certain parts of the environment without having the image target present [2].

4.3.2 Unity3D & Visual Studio

Our main development tool for creating objects for our mixed reality environment on the HoloLens was Unity version 5.6.3 p2 (patch 2). The shapes and outline of our buttons, the visible text placed in free space, and other assets were made available to us through unity. For this project, the user interface (UI) element was taken care of through unity development tool. Visual Studio works in concert with

the Unity development tool as our script editor. This means that when it came to writing code for specific objects to perform an action, all of that was taken care of on the visual studio IDE.

Gestures:
Through Unity development tool, and visual studio, we are able to write scripts that help detect and respond to specific gestures. A gaze manager script tracks your gaze by casting a ray into your immediate environment to detect what you are focusing on. If your focus happens to be on a hologram, and you attempt to use an already created gesture on that object, an event recognizer is called, and it captures your gesture and the required response is carried out. Below in figure 5 are the notable gestures used on the HoloLens.

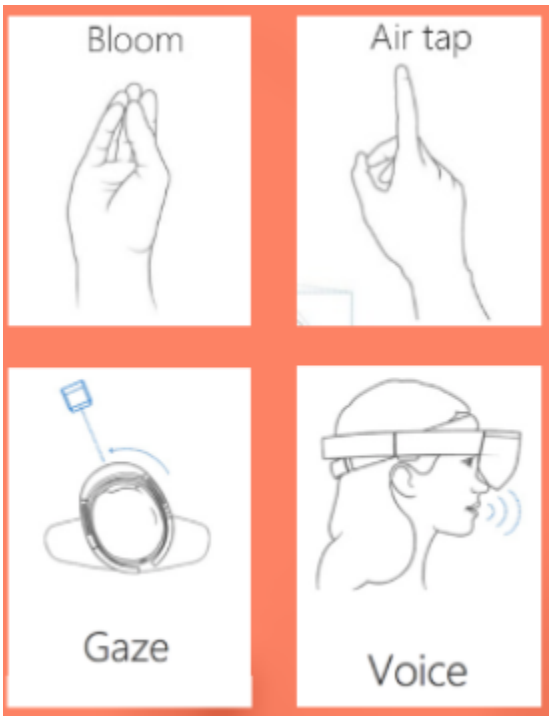


Fig 6: Gestures

Building & Deployment:
After placing the objects you want in your scene, you build the project in unity for the windows store as shown in figure 6. This creates a solution file which is then opened in visual studio, and you can deploy from there [3].

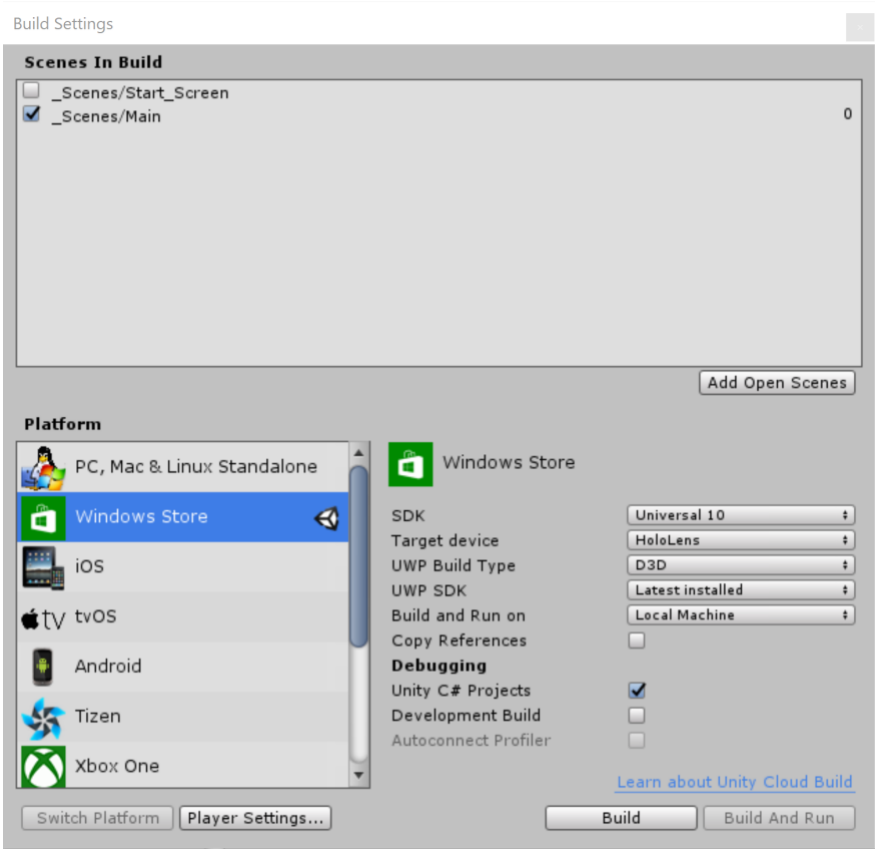


Fig 7: Building Unity Project

4.4 Communication Protocol [IO]

In order to interact with other objects, communication is important. Through a server-client connection, messages could be sent to another device to start up a specific instruction.

Technical Details

From the image below, notice my client function is divided into two parts, a connect function, and a sending function. The connection function of the client connects to a server through an IP address and a port number. Have in mind, server must be started first before client attempts to connect to it. Also, an internet connection is required for this connection to be made. The sending function of the client document simply takes in a value and sends that to the server. Notice that both functions make use of async, and await. This is as a result of asynchronous programming being used in this script. Asynchronous programming is best used when dealing with blocking activities [4].

```
//---------------------------------connection functions----------------------------------------
private async void connect(string host, string port)
{
        try
        {
            if(exchangeTask != null)
            StopExchange();
            clientSocket = new Windows.Networking.Sockets.StreamSocket();
            Windows.Networking.HostName serverHost = new Windows.Networking.HostName(host);
            await clientSocket.ConnectAsync(serverHost, port);
        }
        catch (Exception e)
        {
            errorStatus = e.ToString();
        }
    }

private async void sending(string info)
{
    try
    {

        Stream streamOut = clientSocket.OutputStream.AsStreamForWrite();
        StreamWriter writer = new StreamWriter(streamOut);
        //string request = "Message from Block 4, sent by Block 2";
        await writer.WriteLineAsync(info);
        await writer.FlushAsync();
    }
    catch(Exception e)
    {
        errorStatus = e.ToString();
    }
}
```

Fig 8: Async/Await Connection Functions

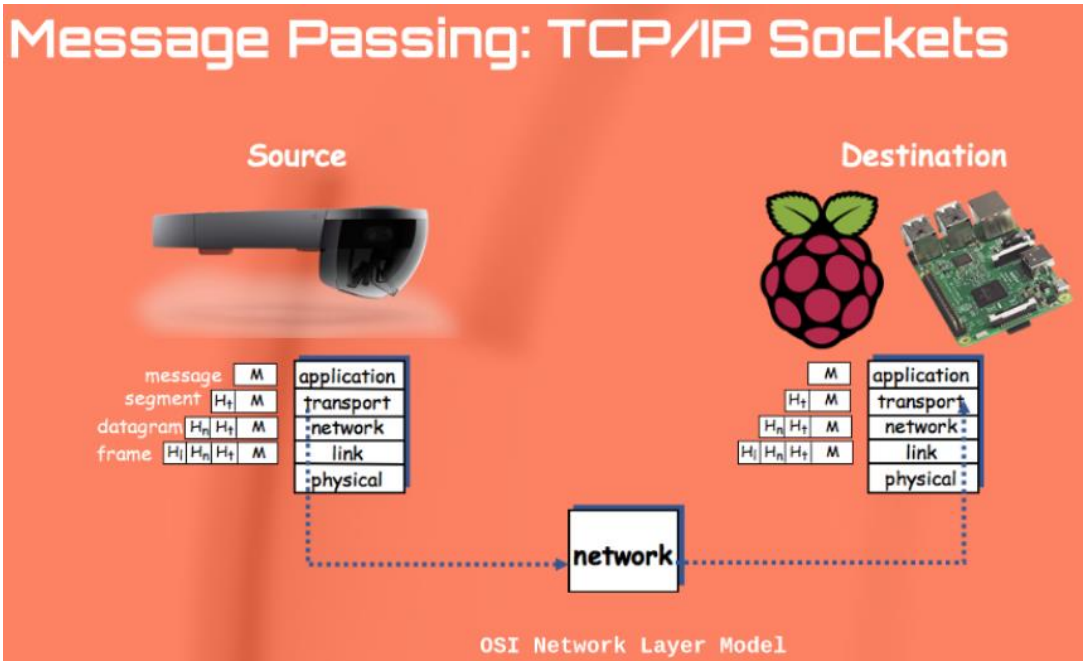


Fig 9: OSI Network Layer Model

4.5 Server Side Architecture [SK]

Fig 10: Raspberry Pi 3 Model B Motherboard

4.5.1 IoT Hub: Raspberry Pi 3 Model B

The Raspberry Pi 3 Model B board was the best fit for this project because of –

1. Quad-Core CPU w/ GPU Architecture
   Allows for effective multitasking. From the very beginning of the project, it was decided that each feature must work independently. Every device runs in its own independent process. Changes can be made by sending the process a message or keyboard signal from the <signal.h> library.

2. Linux Operating System: Raspbian
   The team has a vast amount of experience working with Linux systems. Raspbian connected hardware can be programmed in C/C++ using wiringPi library or in Python. File handling is relatively easy. 'Windows 10 IoT Core' operating system was also considered for its compatibility with other Microsoft products (HoloLens). However, this was not practical for our design.

3. 40-pin GPIO
   Can directly power two 3.3V devices and two 5V devices. Allows for PWM, analog signals, and I^2C protocol communication. For the IoT hub, two 3.3V pins are used to power the sensors and one 5V pin to power the LCD display.

4.5.2 Multithreading

Devices must not interfere with each other, the music player, or the sensor data monitoring operation. Initially the program utilized pthreads library to explicitly parallelize functions of the IoT hub but this was tedious and prone to errors.
In the end product, the Linux software 'Screen' was employed to implicitly manage the execution of processes/threads.
In C, the exec() family of functions was used to run Python scripts.
In Python, the system() function was used to execute system commands.

4.5.3 Status Display LCD

Fig 11: LCD Display

To ensure independent operation and troubleshooting, an LCD character display offers a quick way to view sensor readings, clock, incoming socket messages, device actions, and potential errors. The system uses a standard 16x2 character LCD display with I^2C communication capability.
When a client connects to the system, the screen displays the number of clients until it is used by another function such as sensor update or date/time update.

4.5.4 Sensors

1. **Adafruit Si7021 Temperature and Humidity Sensor**
   Always running in the background and logging the temperature and relative humidity every 5 seconds in a .csv file. Output is displayed on the status LCD. Data can also be read by the sensor functions in the hardware controller code and subsequently sent to the client device for further use.

   Subsystem Details (Appendix D)
   <u>Precision Relative Humidity Module</u>
   Precision: ± 3% RH (max), 0–80% RH
   Operating Range: 0 to 100% RH

   <u>High Accuracy Temperature Module</u>
   Precision: ±0.4 °C (max), –10 to 85 °C
   Operating Range: Up to –40 to +125 °C

   <u>System-On-Chip</u>
   Operating Voltage: 1.9 to 3.6 V
   Power Consumption: 150 µA active current, 60 nA standby current

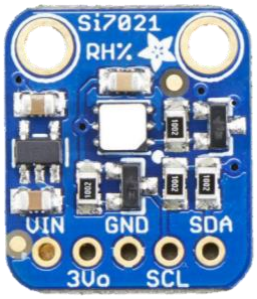


Fig 12: Adafruit Si7021

2. **Adafruit TSL2561 Digital Luminosity/Lux/Light Sensor**
   Approximates human eye response. Constantly running in background and logging the room's visible light brightness every 5 seconds. Also measures IR light level but this value is not used for our project.

   Programmed to toggle lamp on/off if the reading changes by at least 300 Lux.
   Specifications (Appendix E)

Temperature range:
-30 to 80 *C

Dynamic range (Lux):
0.1 to 40,000 Lux

Voltage range:
2.7-3.6V

Fig 13: Adafruit TSL2561

4.5.5 Analog Appliances & IoT Relays

If a device has an on/off switch, it can be converted to an IoT device using the raspberry pi hub. As a cost and time saving measure, existing electrical appliances owned by SEC were converted into IoT devices. These were – a desk lamp, a desk fan, and a blender.

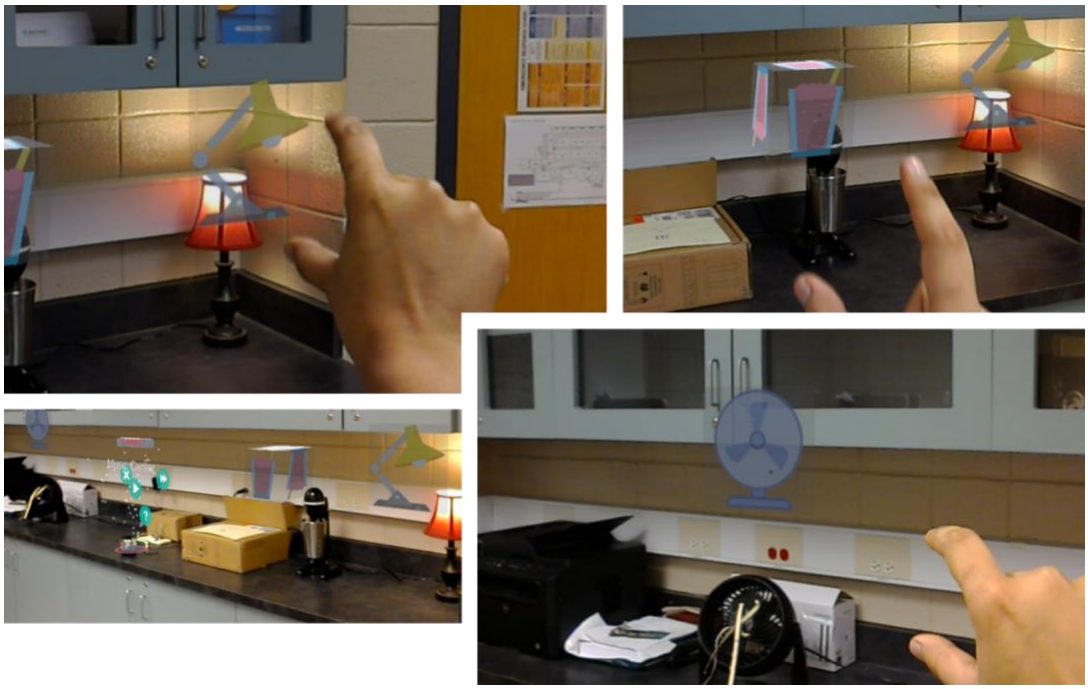


Fig 14: Interacting with Analog Appliances

This goal presented the team with two of our hardest design challenges –

1. Handling the connection to the mains outlet (Dangerous! ☠)
2. Signaling a device with no in-built communication features.

Solution: **IoT Relay by Digital Loggers** (Appendix C)
Each unit can independently control one device, using one GPIO pin. We linked three of them together and plugged the last one into the mains outlet.

4.5.6 Music Player

Using the Raspbian default media player ‘omxplayer’ running in headless mode in the background, any saved song can be accessed and played by the music player. Essentially the music player is just a collection of functions in the hardware controller script. Every song is played in its own new process to ensure parallel operation of all devices.

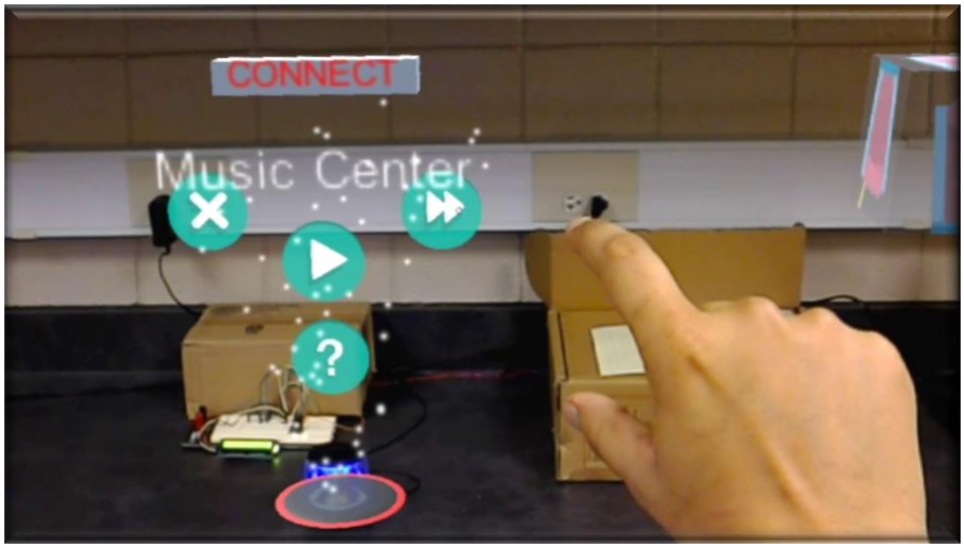


Fig 15: Music Player

Features:

- Actions: Play, Stop, Next, Random.
- Avoids song overlap. This can happen if multiple clients try to play a song at the same time.
- Speech feedback messages for other functions can still be played concurrently.
- If a client gets disconnected, the music player will stop their song automatically.

4.5.7 Speech Feedback

Using omxplayer, prerecorded messages can be played for each action. For speech synthesis, a Mac OS command line program named ‘say’ was used.

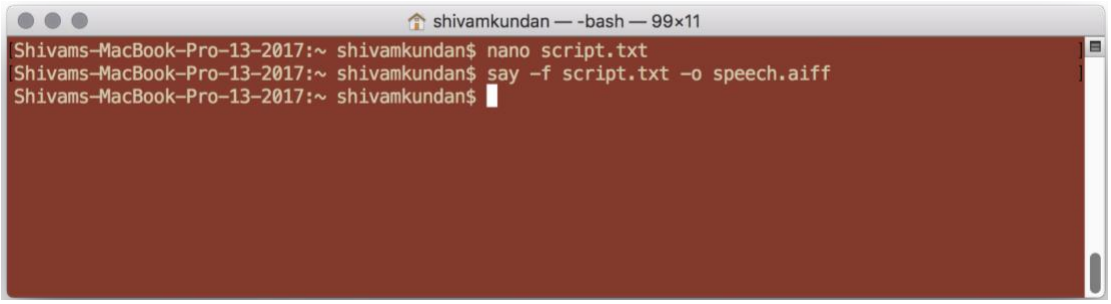
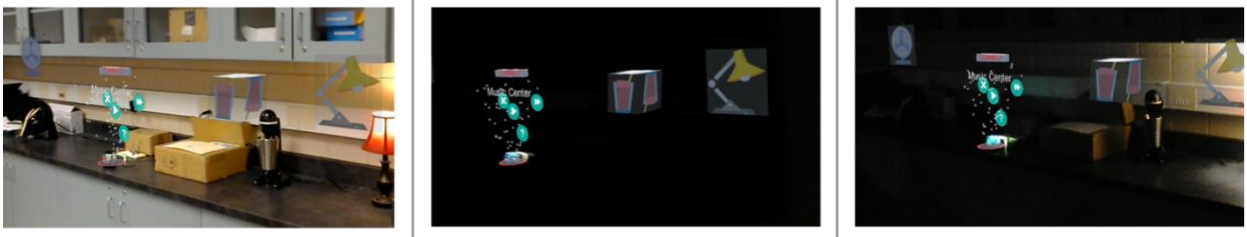


Fig 16: Speech Synthesis on Mac OS Terminal

4.5.8 Automated Room Control (Smart Services)

Using the sensors, the Smart Room can detect when a specific activity occurs and then take an action such as playing a message or turning a device on/off. If needed, it is able to cross reference the readings with data from the internet such as weather and traffic.

Using the TSL2561 Lux sensor, if the system detects a change of more than 300 Lux, it toggles the desk lamp on or off and plays the corresponding speech feedback.



1. Normal Lighting | 2. Lights Off | 3. Lamp turns on automatically (Delay: 1 to 5 seconds)

Fig 17: Automated Night Light

Date	Time	Temp (C)	Humidity	Brightness
11/26/17	3:34:53PM	24.126C	23.41%	739lux
11/26/17	3:34:55PM	24.126C	23.41%	740lux
11/26/17	3:34:57PM	24.126C	23.41%	740lux

Table 1: Sample of Sensor Log File

Appendix H mentions the possibility of making a non-intrusive monitoring system using these two sensors.

5. Future Expansions [SK]

5.1 Extended Gestures

HoloLens offers the use of three gestures: gaze, tap, and bloom. Using some extra hardware for the raspberry pi, gestures can be extended to be swipes, pinches, etc.
Links:
Hover Adds Gesture Control to Arduino and Raspberry Pi Projects
Turn on a Lamp with a Gesture-Controlled Harry Potter Wand

5.2 Artificial Intelligence

Amazon's AI assistant 'Alexa' is open source and can be integrated into a wide variety of raspberry pi projects. She can also be made to execute custom programs.
Links:
http://www.cyber-omelette.com/2016/11/alexa-pi.html
http://www.cyber-omelette.com/2017/01/alexa-run-script.html

5.3 More Sensors

Adafruit 9-DOF Accel/Mag/Gyro+Temp Breakout Board - LSM9DS0:
9-degrees-of-freedom sensor board
Link: https://www.adafruit.com/product/3463

I2CThermal Camera:
Link: https://www.adafruit.com/product/3538

5.4 Motion Tracker/Controller

Track motion in extremely high detail.
https://www.adafruit.com/product/2106

6. Development Tools for Future Projects [AP]

In this section we will discuss the hardware used during the completion of the project. We will also give possible alternatives, so other hardware can be used and strict guidelines of the system requirements are not abused. First the system requirements, given by Microsoft, in order to use the Hololens Emulator as well as to transfer programs from your computer hardware to the Hololens hardware. The Hololens Emulator requires Hyper-V capabilities and uses RemoteFx for hardware accelerated graphics. In order to use this emulator your PC is required to meet these standards:

- 64-bit Windows 10 Pro, Enterprise, or Education (**The Home edition does not support Hyper-V or the HoloLens emulator**)
- 64-bit CPU
- CPU with 4 cores (or multiple CPU's with a total of 4 cores)
- 8 GB of RAM or more
- In the BIOS, the following features must be supported and enabled:
  - Hardware-assisted virtualization
  - Second Level Address Translation (SLAT)
  - Hardware-based Data Execution Prevention (DEP)
- GPU (The emulator might work with an unsupported GPU, but will be significantly slower)
  - DirectX 11.0 or later
  - WDDM 1.2 driver or later
- The technological specifications of the PC that was used during the course of the project were as follows:

Processor	2.6 GHz Intel Core i7
RAM	16 GB DDR3
Memory Speed	1600 MHz
Hard Drive	60 GB flash memory solid state
Graphics Coprocessor	NVIDIA GTX960M 2G GDDR5
Card Description	dedicated
Processor Count	4

Table 2: System Requirements

If the PC one has in their possession does not meet these requirements, then one will need to be acquired before work can be done with the Hololens. Because the PC that

was used for majority of the project was the personal computer of one of the members of the team, it was priced at over $1000. We realize that this is not a feasible budget for some people. There are a few already assembled computers that meet these requirements and costs are significantly lower. In order to find one that fits your budget, one must go online and search for PCs by price as well as specified specifications. Here are a few that we found had the minimal specs, but still got the job done.

2017 HP Elite 8300 Small Form Factor Desktop Computer

- Memory
  - 16GB DDR3 RAM
- Processor
  - Intel Quad Core i7-3770 3.4GHz Processor
- Price – $450.00

2017 HP Elite 8300 Small Form Factor Desktop Computer

- Memory
  - 16GB DDR3 RAM
- Processor
  - Intel Quad Core i7-3770 3.4GHz Processor
- Price – $824.66

HP ENVY Desktop Computer

- Memory
  - 16 GB DDR4-2133 SDRAM
- Processor
  - Intel Quad Core i7-7700
- Price – $1,159.44

There is a possibility that the PC one would like to use has meet some of the technical requirements, but not all of them. So, here are some PC parts that you could integrate into the PC one has already.

Spec	Part	Price
Quad- Core Processor	AMD Athlon X4 845 and Near-Silent 95W AMD Thermal Solution AD845XACKASBX	$55.91
	AMD FX 4-Core Black Edition FX-4300, FD4300WMHKBOX	$69.95
	AMD A8-7600 Quad-Core 3.1 GHz Socket FM2+ 65W Desktop Processor AMD Radeon R7 (AD7600YBJABOX)	$71.83
RAM (8GB)	604506-B21 RAM Module - 8 GB (1 x 8 GB) - DDR3 SDRAM	$30.20
	8GB (1x8GB) Dual Rank x4 PC3-10600 (DDR3-1333) Registered CAS-9 Memory Kit	$29.48
	Kingston 8 GB DDR3 SDRAM Memory Module 8 GB (1 x 8 GB) 1333MHz DDR31333/PC310600 ECC DDR3 SDRAM 240pin DIMM KTH-PL313/8G	$52.29
Windows 10 must be used as the operating system when trying to complete this project. The specific types of this operating system (Pro, Enterprise, Education) can we acquired through the school or organization that is sponsoring this project.		

Table 3: Specifications

7. Summary [SK]

The HoloLens IoT Smart Room system monitors real-time data from high precision sensors and then overlays them in a way that enables more efficient use of the information. Combined with the ability to control physical devices, several Smart Services can be set up for each individual user or groups of users.

At a cost of only $90, the Smart Room is an effective proof of concept for projects involving AR/VR on the HoloLens. Because of its uniqueness and open-ended nature, the project was extremely research intensive. We hope our documentation and code will be helpful for future teams attempting similar projects.

8. References

[1] Statista, "Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions)," 1 November 2016. [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. [Accessed 9 December 2017].

[2] Vuforia, "Developing Vuforia Apps for HoloLens," Vuforia, [Online]. Available: https://library.vuforia.com/articles/Training/Developing-Vuforia-Apps-for-HoloLens. [Accessed 9 December 2017].

[3] "Unity development overview," [Online]. Available: https://developer.microsoft.com/en-us/windows/mixed-reality/unity_development_overview.

[4] "Asynchronous programming with async and await (C#)," 22 May 2017. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/index. [Accessed 9 December 2017].

[5] Digital Loggers Inc, [Online]. Available: https://dlidirect.com/products/iot-power-relay. [Accessed 9 December 2017].

[6] L. Klint, "Holographic programming: a HoloLens how-to in a mixed reality world," 27 May 2016. [Online]. Available: https://www.pluralsight.com/blog/software-development/holographic-programming-hololens. [Accessed 31 March 2017].

[7] NAE, [Online]. Available: http://www.engineeringchallenges.org/challenges.aspx. [Accessed 31 March 2017].

[8] T. I. Paul Hockett, "Augmented Reality with Hololens: Experiential Architectures Embedded in the Real World," *Arxiv,* 13 October 2016.

[9] A. S. M. G. F. L. Giovanni Piumatti, "Spatial Augmented Reality meets robots: Human-machine interaction in cloud-based projected gaming environments," in *Consumer Electronics (ICCE) 2017 IEEE International Conference*, 2017.

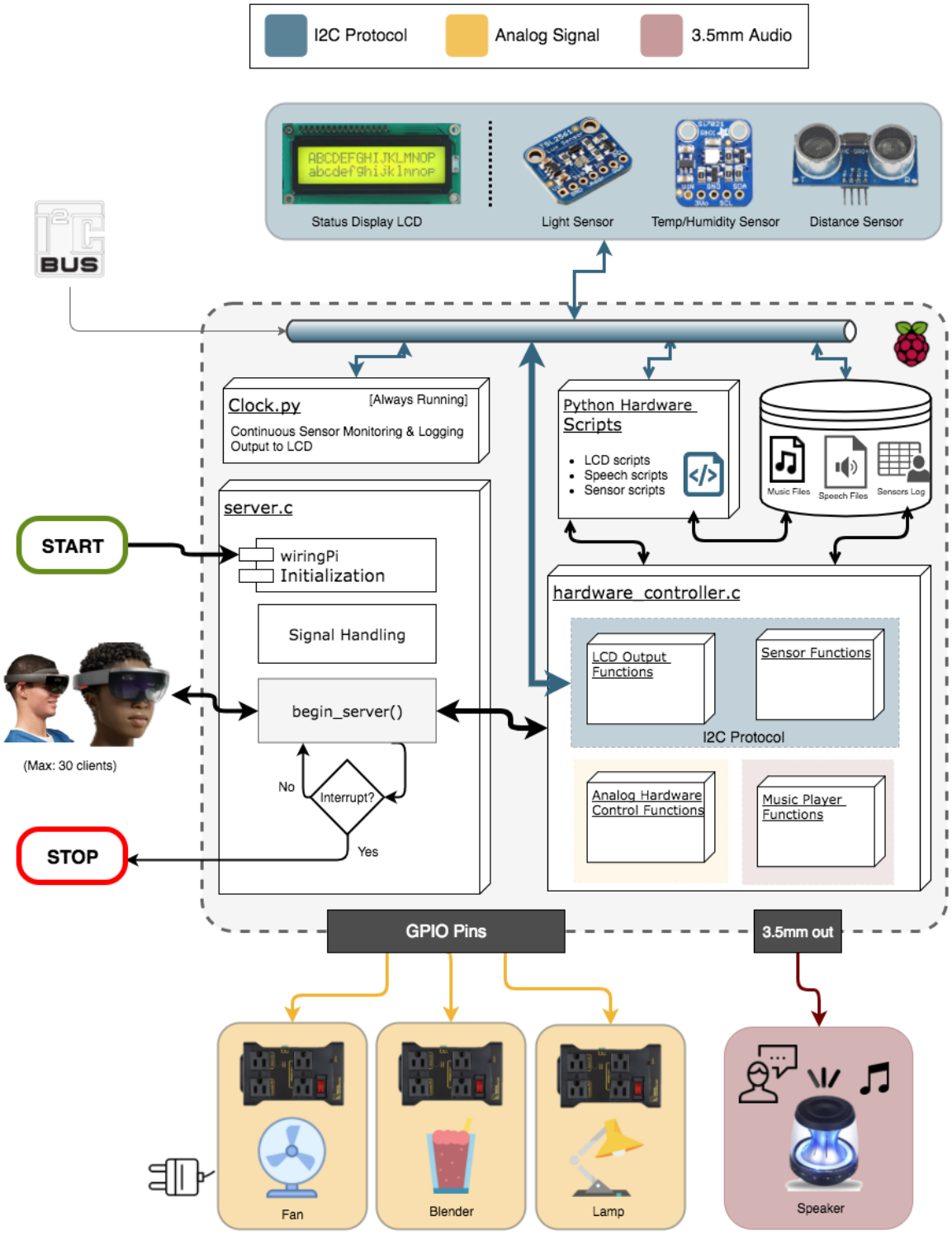
[10] [Online]. Available: https://i.ytimg.com/vi/DilzwF90vec/maxresdefault.jpg.

[11] [Online]. Available: https://d.ibtimes.co.uk/en/full/1420390/how-microsoft-hololens-will-enable-scientists-work-virtually-mars.jpg.

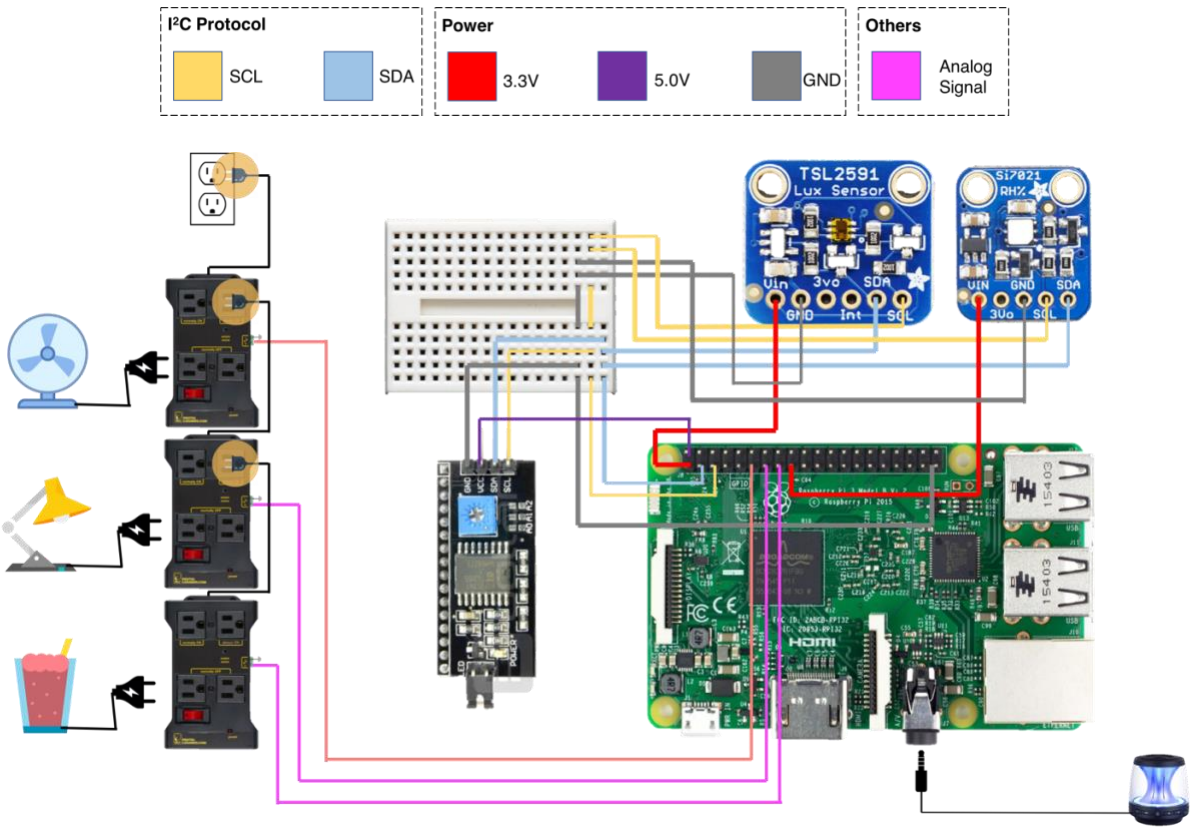
[12] B. Stackpole, "IoT-and-augmented-reality-A-match-made-in-heaven," 1 November 2016. [Online]. Available: http://internetofthingsagenda.techtarget.com/feature/IoT-and-augmented-reality-A-match-made-in-heaven. [Accessed 9 December 2017].

9. Appendices [SK] [KN]

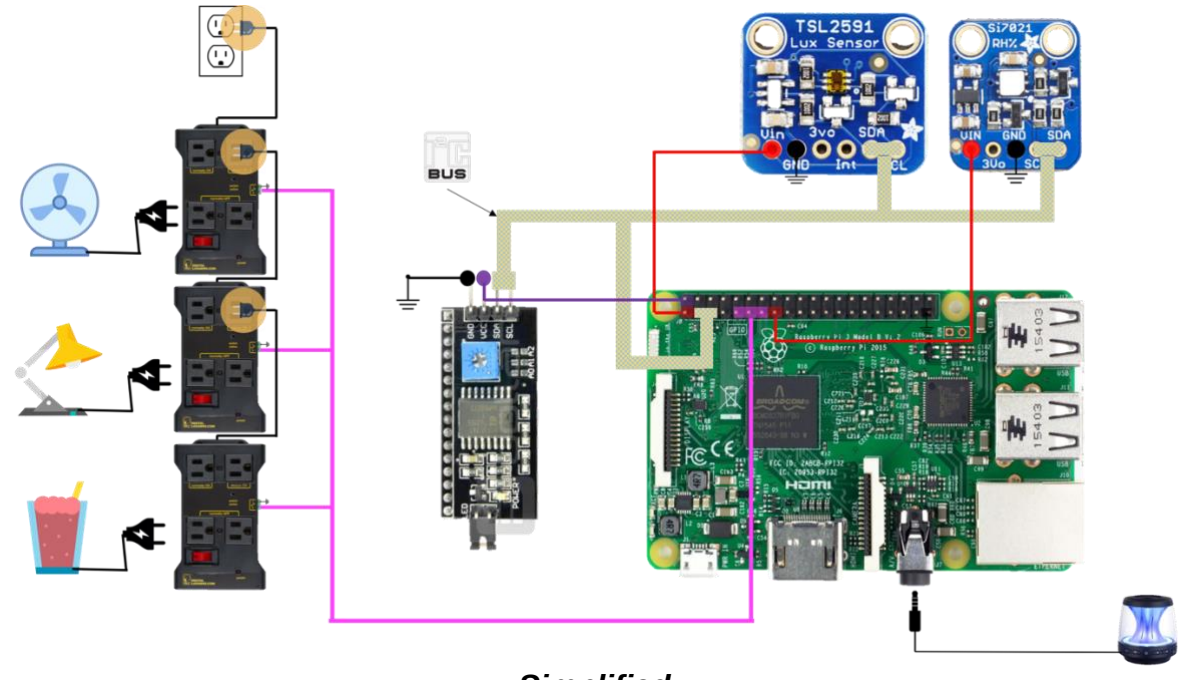
9.1 Appendix A: IoT Hub Operation Diagram [SK]



9.2 Appendix B: IoT Hub Wiring Diagrams [SK]

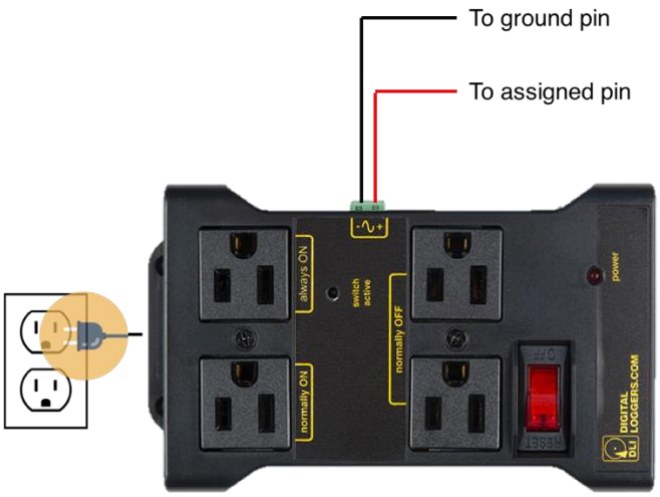


Detailed

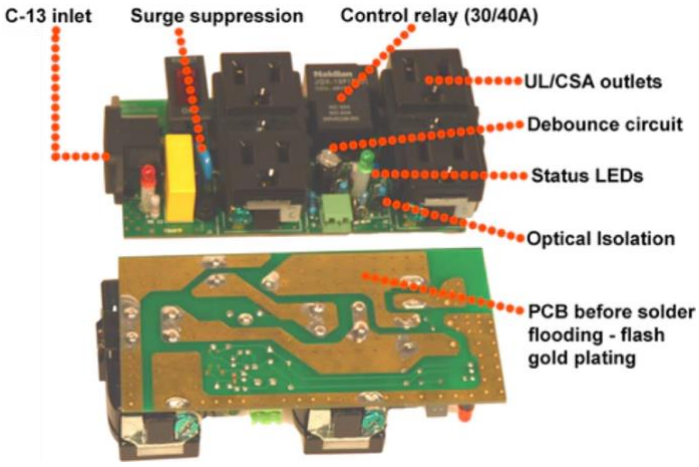


Simplified

9.3 Appendix C: IoT Electrical Relay Operation [SK]



Wiring Diagram

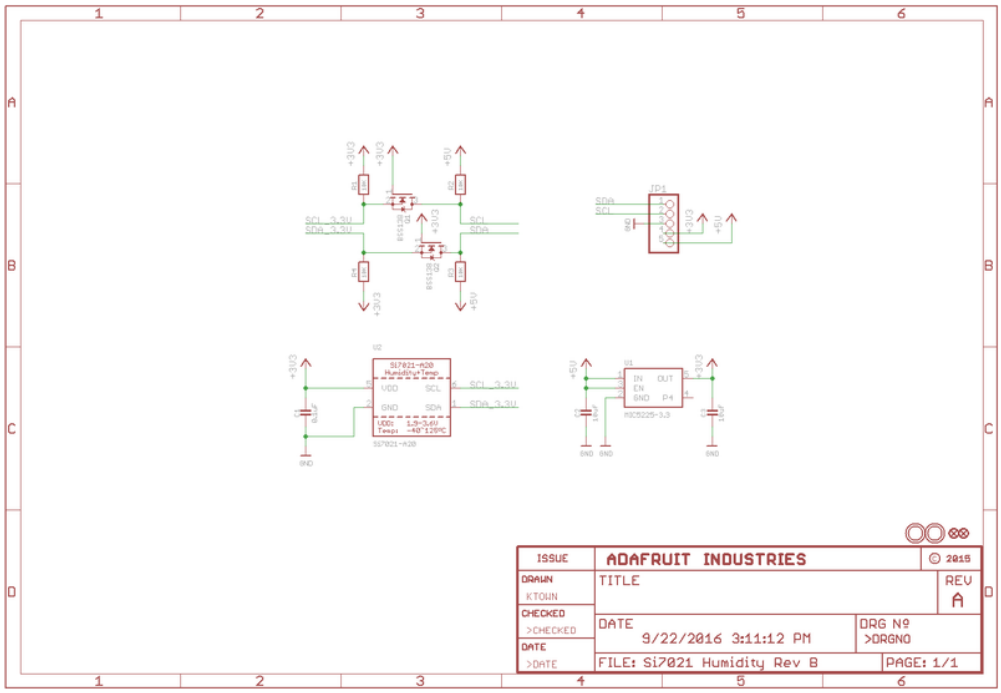


Safety Features [5]

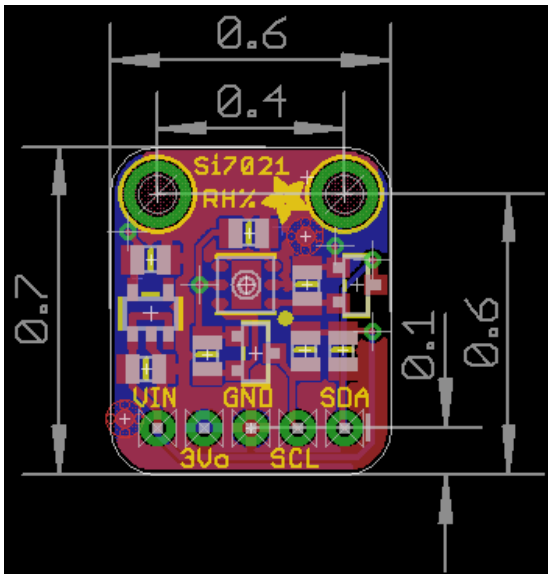
Link to manufacturers webpage: https://dlidirect.com/products/iot-power-relay

9.1 Appendix D: Adafruit Si7021 Temperature + Humidity Sensor Board [SK]

Purpose: Get the current ambient temperature and humidity
Datasheet:https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf



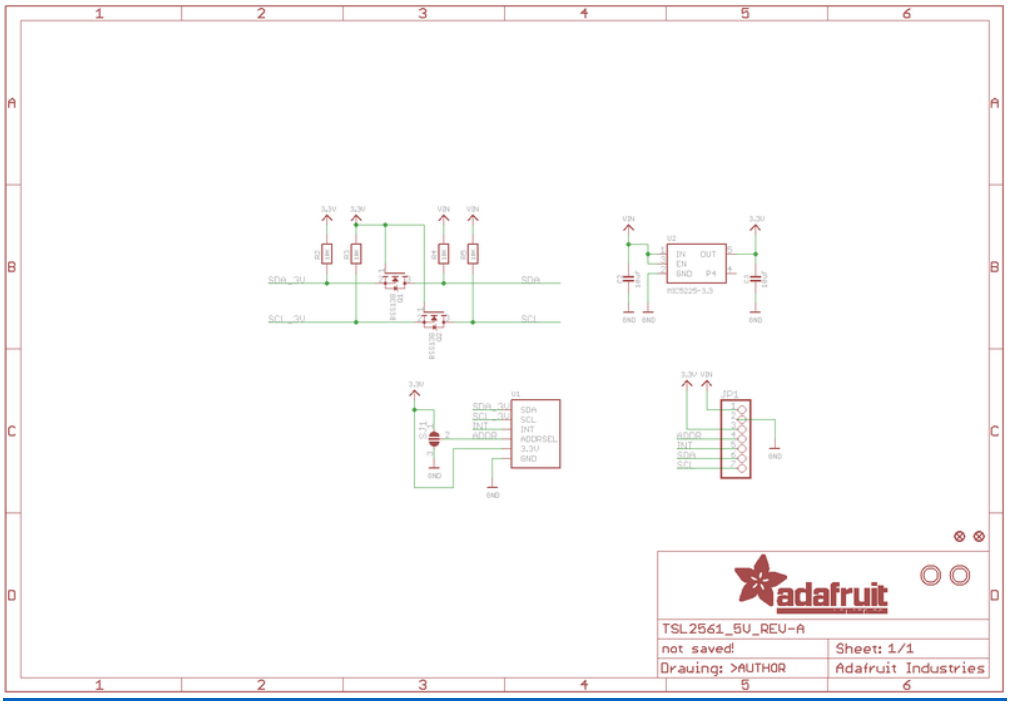
Schematic



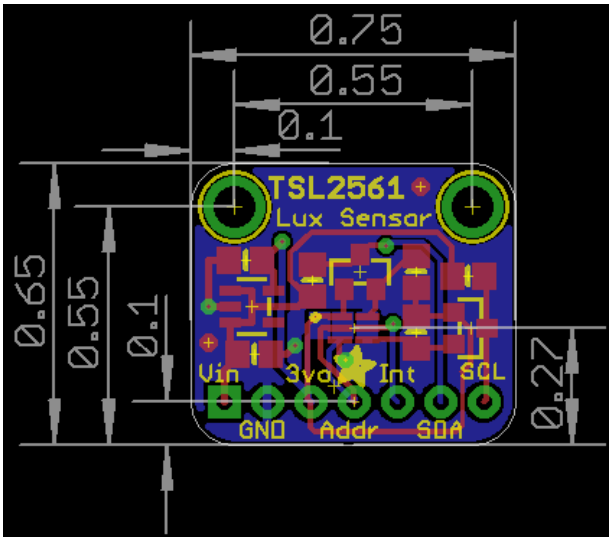
Fabrication Print

9.2 Appendix E: Adafruit TSL2561 Digital Luminosity Sensor Board [SK]

Purpose: Low power, digital luminosity (light) sensor
Datasheet: https://cdn-shop.adafruit.com/datasheets/TSL2561.pdf

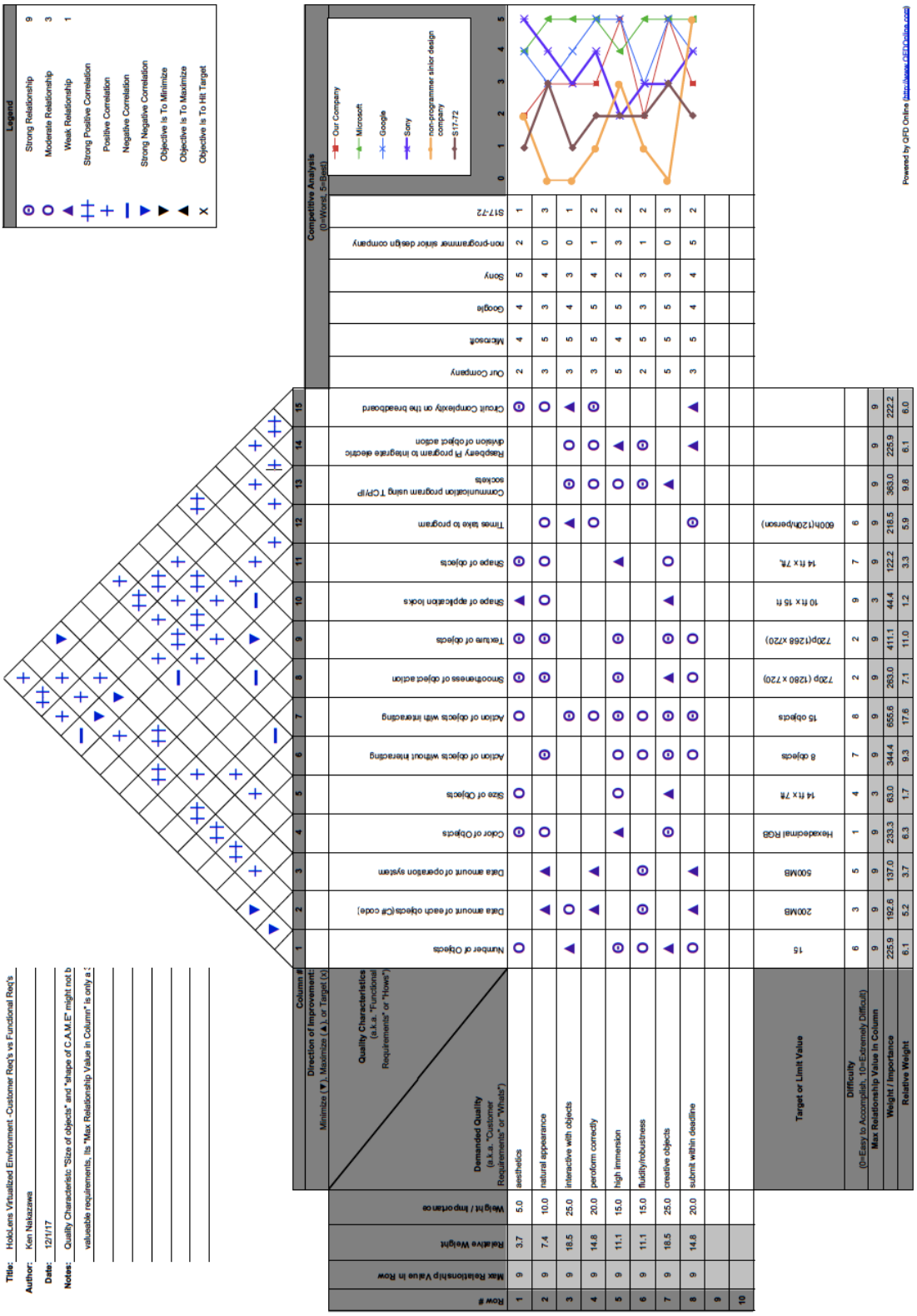


Schematic



Fabrication Print

9.3 Appendix F: House of Quality [KN]



Row #	Max Relationship Value in Row	Relative Weight	Weight / Importance	Demanded Quality (a.k.a. "Customer Requirements" or "Whats")	Column #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
				Quality Characteristics (a.k.a. "Functional Requirements" or "Hows")	Direction of Improvement: Minimize (▼), Maximize (▲), or Target (x)	Number of Objects	Data amount of each objects(C# code)	Data amount of operation system	Color of Objects	Size of Objects	Action of objects without interacting	Action of objects with interacting	Smoothness of object action	Texture of objects	Shape of application looks	Shape of objects	Times take to program	Communication program using TCP/IP sockets	Responsery Pf program to integrate electric division of object action	Circuit Complexity on the breadboard
1	9	3.7	5.0	aesthetics		○			◎	○		○	○	◎	▼	◎	◎			◎
2	9	7.4	10.0	natural appearance			▲	▲	○		◎		◎	◎	▼	○	○			○
3	9	18.5	25.0	interactive with objects		▲	○					◎					◎	◎	▲	◎
4	9	14.8	20.0	perform correctly			▲	▲				○					○	◎	○	◎
5	9	11.1	15.0	high immersion		◎			▲	○	○	◎	◎		▼	▲		○	▼	▲
6	9	11.1	15.0	fluidity/robustness		○	◎	◎			○	○						◎	◎	
7	9	18.5	25.0	creative objects		▲			◎	▲	◎	◎	▲	◎	▲	○		▲		
8	9	14.8	20.0	submit within deadline		○	▲	▲			○	◎	○	○			◎		▲	▲
9	9																			
10																				
				Target or Limit Value		15	200MB	500MB	Hexadecimal RGB	14 ft x 7ft	8 objects	15 objects	720p (1280 x 720)	720p(1268 x720)	10 ft x 15 ft	14 ft x 7ft	600ft(12ch/person)			
				Difficulty (0=Easy to Accomplish, 10=Extremely Difficult)		6	3	5	1	4	7	8	2	2	9	7	6			
			9	Max Relationship Value in Column		9	9	9	9	3	9	9	9	9	3	9	9	9	9	9
			225.9	Weight / Importance		9	192.6	137.0	233.3	63.0	344.4	655.6	283.0	411.1	44.4	122.2	216.5	363.0	225.9	222.2
			6.1	Relative Weight		6.1	5.2	3.7	6.3	1.7	9.3	17.6	7.1	11.0	1.2	3.3	5.9	9.8	6.1	6.0

Row #	Max Relationship Value in Row	Relative Weight	Weight / Importance	Demanded Quality (a.k.a. "Customer Requirements" or "Whats")	Column #	Our Company	Microsoft	Google	Sony	non-programmer sinior design company	S17-72
				Quality Characteristics (a.k.a. "Functional Requirements" or "Hows")	Direction of Improvement: Minimize (▼), Maximize (▲), or Target (x)						
1	9	3.7	5.0	aesthetics		2	4	4	5	2	1
2	9	7.4	10.0	natural appearance		3	5	3	4	0	3
3	9	18.5	25.0	interactive with objects		3	5	4	3	0	1
4	9	14.8	20.0	perform correctly		3	5	5	4	1	2
5	9	11.1	15.0	high immersion		5	4	5	2	3	2
6	9	11.1	15.0	fluidity/robustness		2	5	3	3	1	2
7	9	18.5	25.0	creative objects		5	5	5	3	0	3
8	9	14.8	20.0	submit within deadline		3	5	4	4	5	2
9	9										
10											

9.4 Appendix G: Materials List [SK]

#	Vendor	Product	Quantity	Original Price	Total paid	Source
1	Microsoft	HoloLens	1	$3,000	$0	Mr. Howard Lo
2	RPi Foundation	Raspberry Pi 3 Model B	1	$35	$0	Professor Cubley
3	-	LCD Display 16x2	1			Professor Cubley
4	Digital Loggers	IoT Relays	3	$29.92	$89.76	Dr Anagnostopoulos
5	Adafruit	SL2561 Digital Luminosity/Lux/Light Sensor Breakout	1	$9.15	$0	Shivam
6	Adafruit	Si7021 Temperature & Humidity Sensor Breakout Board	1	$9.95	$0	Shivam
7	Honeywell	Desk Fan	1	-	$0	SEC
8	-	Desk Lamp	1	-	$0	SEC
9	-	Blender	1	-	$0	SEC

Total Expenditure: **$89.76**

9.5 Appendix H: Observations from Sensor Data [SK]

Both sensors logged data every 3 to 6 seconds, with the mild variation due to processing/file handling overhead. After running the system continually for four days, a sizeable amount of data was gathered. The purpose of performing this experiment was to see if anything could be learned/inferred from the data.

Total # of readings = 11,814
Frequency = 3 to 6 s between readings

The inferences are based on cross referencing sensor readings with physical observations and information from the internet.

Excerpts from the sensor log and observations are listed on the next two pages.

11/26/17	3:34:53PM	24.126C	23.41%	739lux
11/27/17	9:05:09PM	24.815C	22.43%	685lux
11/27/17	9:05:26PM	24.815C	22.43%	693lux
11/27/17	9:05:43PM	24.815C	22.43%	3lux
11/27/17	9:05:59PM	24.815C	22.43%	1lux
11/27/17	9:06:16PM	24.815C	22.43%	1lux
11/27/17	9:06:33PM	24.815C	22.43%	1lux
11/27/17	10:46:30PM	24.126C	20.96%	1lux
11/27/17	10:46:47PM	24.126C	20.96%	1lux
11/27/17	10:47:03PM	24.126C	20.96%	0lux
11/27/17	10:47:20PM	24.126C	20.96%	0lux
11/28/17	6:29:37AM	23.436C	20.96%	0lux
11/28/17	6:29:54AM	23.436C	20.96%	0lux
11/28/17	6:30:11AM	23.436C	20.96%	0lux
11/28/17	6:30:27AM	23.436C	20.96%	1lux
11/28/17	6:30:44AM	23.436C	20.96%	1lux
11/28/17	6:31:01AM	23.436C	20.96%	1lux

Begin logging data
[3:43:53 PM]

Last person to leave the lab
[9:05:43 PM]

Hallway Lights Turned Off
[10:47:03 PM]

Twilight Detected (?)
[6:30:27 AM]

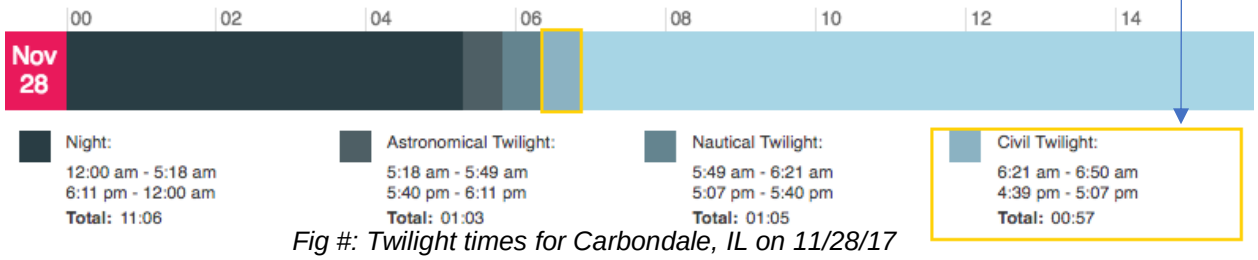


Fig #: Twilight times for Carbondale, IL on 11/28/17

11/28/17	9:00:22AM	23.436C	20.96%	1lux
11/28/17	9:00:39AM	23.436C	20.96%	1lux
11/28/17	9:00:56AM	23.436C	20.96%	605lux
11/28/17	9:01:12AM	23.436C	20.96%	603lux
11/28/17	7:06:58PM	24.815C	23.41%	694lux
11/28/17	7:07:14PM	24.815C	23.90%	429lux
11/28/17	7:07:31PM	24.815C	23.41%	407lux
11/28/17	7:07:48PM	24.815C	23.41%	693lux
11/28/17	7:08:04PM	24.815C	23.41%	642lux
11/29/17	7:49:57PM	24.815C	28.80%	652lux

First person to arrive
[9:00:56 AM]

Person/people standing directly in front of sensors reduces light falling on detector.
[7:07:14 PM]

4 to 6 more people entered the lab in this period. From our physical observation, each person increases humidity by ~1%
[7:08:04 PM]
[7:49:47 PM]

11/29/17	11:11:15PM	24.815C	29.78%	636lux
11/29/17	11:11:17PM	24.815C	29.78%	637lux
11/29/17	11:11:20PM	24.815C	29.78%	0lux
11/29/17	11:11:25PM	24.815C	29.78%	6lux
11/29/17	11:11:28PM	24.815C	29.78%	6lux
11/29/17	11:11:30PM	24.815C	29.78%	6lux
11/29/17	11:11:33PM	24.815C	29.78%	6lux
11/29/17	11:11:36PM	24.815C	29.78%	6lux
11/29/17	11:11:38PM	24.815C	29.78%	6lux
11/29/17	11:11:41PM	24.815C	29.78%	6lux
11/29/17	11:11:44PM	24.815C	29.78%	6lux
11/29/17	11:11:46PM	24.815C	29.78%	6lux
11/29/17	11:11:49PM	24.815C	29.78%	6lux
11/29/17	11:11:51PM	24.815C	29.78%	6lux
11/29/17	11:11:54PM	24.815C	29.78%	6lux
11/29/17	11:11:57PM	24.815C	29.78%	6lux
11/29/17	11:11:59PM	24.815C	29.78%	6lux
11/29/17	11:12:02PM	24.815C	29.78%	6lux
11/29/17	11:12:05PM	24.815C	29.78%	6lux
11/29/17	11:12:07PM	24.815C	29.78%	6lux
11/29/17	11:12:10PM	24.815C	29.78%	6lux
11/29/17	11:12:13PM	24.815C	29.78%	6lux
11/29/17	11:12:15PM	24.815C	29.78%	6lux
11/29/17	11:12:18PM	24.815C	29.78%	691lux
11/29/17	11:12:23PM	24.815C	29.78%	659lux
11/29/17	11:12:25PM	24.815C	29.78%	654lux
11/29/17	11:12:28PM	24.815C	29.78%	649lux
11/30/17	12:39:05PM	24.815C	26.84%	642lux
11/30/17	12:39:16PM	24.815C	26.84%	630lux
11/30/17	12:39:28PM	24.815C	26.84%	104lux
11/30/17	12:39:41PM	24.815C	27.33%	26lux
11/30/17	12:39:53PM	24.815C	27.33%	501lux
11/30/17	12:40:07PM	24.815C	29.29%	638lux
11/30/17	12:40:19PM	24.815C	28.80%	640lux
11/30/17	12:40:30PM	24.815C	27.82%	640lux

<u>Testing Smart Service:</u>
Automated Night Light
[11:11:17 PM]

1. Lux<100 detected
   [11:11:20 PM]
2. Small processing delay
   (Have to wait 3 to 6 seconds for next readings)

Desk Lamp turns on automatically
[11:11:25 PM]

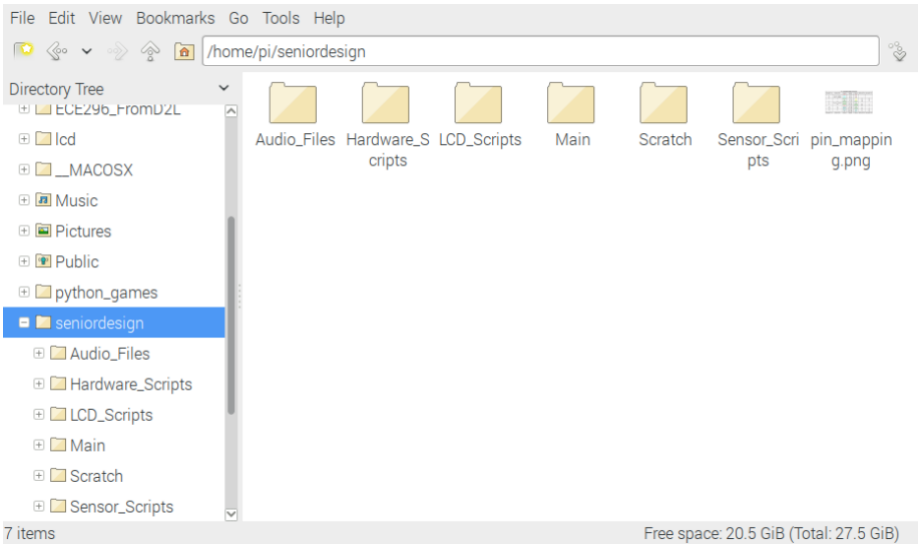
~5s delay

Desk lamp will turn off automatically but it cannot be confirmed from this data alone.
[11:12:18 PM]

Someone was likely checking out our sensor and LCD setup. This reduces the light reaching the sensor's receiver, causing these dips in readings.
[12:39:28 PM]
[12:40:07 PM]

39 seconds

9.6 Appendix I: IoT Hub Code [SK]



Organization of Source Files

Main/server.c
Maintains communication with HoloLens

```
1.  //Shivam Kundan
2.  //S17-71-HOL1
3.  //Fall 2017
4.
5.  /*This program -
6.    Handles multiple HoloLens socket connections.
7.    Sends message to hardware controller.
8.    Sends message back to client.
9.    Socket programming code modified from: goo.gl/mPWPbP
10.   */
11.
12. #include <stdio.h>
13. #include <string.h>                  //strlen
14. #include <stdlib.h>
15. #include <errno.h>
16. #include <unistd.h>
17. #include <arpa/inet.h>
18. #include <sys/types.h>
19. #include <sys/socket.h>
20. #include <netinet/in.h>
21. #include <sys/time.h>                //FD_SET, FD_ISSET, FD_ZERO macros
22. #include <wiringPi.h>
23. #include <wiringPiI2C.h>
24. #include <signal.h>
25. #include <sys/wait.h>
26. #include <time.h>
27. #include <math.h>
28. #include "mappings.h"               //For misc values used in the code
29. #include "hardware_controller.c"    //Contains all hardware control functions
30.
31. #define PORT 51717
32.
33. void begin_server();               //Maintains communication with hololens(s)
```

```
34. void control_hardware(int data);   //Sends received integers to hardware control functi
    ons
35.
36. int main(int argc , char *argv[])
37. {
38.     //Initialize GPIO pins using wiringPi mapping
39.     wiringPiSetup();
40.     pinMode (DESK_FAN_PIN, OUTPUT);
41.     pinMode (LAMP_PIN, OUTPUT);
42.     pinMode(BLENDER_PIN,OUTPUT);
43.     pinMode (outpin, OUTPUT);
44.     pinMode (inpin, INPUT);
45.
46.     //Signal handling
47.     signal(SIGINT,sig_handler);
48.
49.     //Will be used later for random song function
50.     srand(time(NULL));
51.
52.     //Display and play welcome message
53.     system("screen -dm bash -c \"omxplayer -
    o local /home/pi/seniordesign/Audio_Files/Speech/system_on.mp3\"");
54.     system("python  /home/pi/seniordesign/LCD_Scripts/lcd_clear.py");
55.     system("python  /home/pi/seniordesign/LCD_Scripts/lcd_systemon.py");
56.
57.     begin_server();
58.
59.     return 0;
60. }
61.
62. void begin_server()
63. /*Maintains communication with one or more hololens. Sends received data to
64.   hardware controlling functions*/
65. {
66.     int opt = 1;
67.     int master_socket , addrlen , new_socket , client_socket[30] ,
68.           max_clients = 30 , activity, i , valread , sd;
69.     int max_sd;
70.     struct sockaddr_in address;
71.     char buffer[1025];  //data buffer of 1K
72.
73.     //set of socket descriptors
74.     fd_set readfds;
75.
76.     //initialise all client_socket[] to 0 so not checked
77.     for (i = 0; i < max_clients; i++) {
78.         client_socket[i] = 0;
79.     }
80.
81.     //create a master socket
82.     if( (master_socket = socket(AF_INET , SOCK_STREAM , 0)) == 0) {
83.         perror("socket failed");
84.         exit(EXIT_FAILURE);
85.     }
86.
87.     //set master socket to allow multiple connections ,
88.     if( setsockopt(master_socket, SOL_SOCKET, SO_REUSEADDR, (char *)&opt,
89.           sizeof(opt)) < 0 ) {
90.         perror("setsockopt");
91.         exit(EXIT_FAILURE);
92.     }
```

```
93.
94.     //type of socket created
95.     address.sin_family = AF_INET;
96.     address.sin_addr.s_addr = INADDR_ANY;
97.     address.sin_port = htons( PORT );
98.
99.     //bind the socket to localhost port PORT
100.            if (bind(master_socket, (struct sockaddr *)&address, sizeof(address))<0) {

101.                perror("bind failed");
102.                exit(EXIT_FAILURE);
103.            }
104.            printf("Listener on port %d \n", PORT);
105.
106.            //try to specify maximum of 3 pending connections for the master socket
107.            if (listen(master_socket, 3) < 0) {
108.                perror("listen");
109.                exit(EXIT_FAILURE);
110.            }
111.
112.            //accept the incoming connection
113.            addrlen = sizeof(address);
114.            puts("Waiting for connections ...");
115.
116.
117.            while(1)
118.            {
119.                //clear the socket set
120.                FD_ZERO(&readfds);
121.
122.                //add master socket to set
123.                FD_SET(master_socket, &readfds);
124.                max_sd = master_socket;
125.
126.                //add child sockets to set
127.                for ( i = 0 ; i < max_clients ; i++) {
128.                    //socket descriptor
129.                    sd = client_socket[i];
130.
131.                    //if valid socket descriptor then add to read list
132.                    if(sd > 0)
133.                        FD_SET( sd , &readfds);
134.
135.                    //highest file descriptor number, need it for the select function
136.                    if(sd > max_sd)
137.                        max_sd = sd;
138.                }
139.
140.                //wait for an activity on one of the sockets , timeout is NULL ,
141.                //so wait indefinitely
142.                activity = select( max_sd + 1 , &readfds , NULL , NULL , NULL);
143.
144.                if ((activity < 0) && (errno!=EINTR))
145.                {
146.                    printf("select error");
147.                }
148.
149.                //If something happened on the master socket ,
150.                //then its an incoming connection
151.                if (FD_ISSET(master_socket, &readfds))
152.                {
```

```
153.                    if ((new_socket = accept(master_socket,
154.                            (struct sockaddr *)&address, (socklen_t*)&addrlen))<0)
            {
155.                        perror("accept");
156.                        exit(EXIT_FAILURE);
157.                    }
158.
159.                    //inform user of socket number - used in send and receive commands

160.                    printf("New connection , socket fd is %d , ip is : %s , port : %d \n
    " , new_socket , inet_ntoa(address.sin_addr) , ntohs(address.sin_port));
161.
162.                    //add new socket to array of sockets
163.                    for (i = 0; i < max_clients; i++) {
164.                        //if position is empty
165.                        if( client_socket[i] == 0 ) {
166.                            client_socket[i] = new_socket;
167.                            printf("Adding to list of sockets as %d\n" , i);
168.
169.                            //Play a welcome message
170.                            system("screen -dm bash -c \"omxplayer -
    o local /home/pi/seniordesign/Audio_Files/Speech/client_connected.mp3\"");
171.
172.                            //Display number of clients connected on LCD display
173.                            lcd_client_connected(i);
174.                            break;
175.                        }
176.                    }
177.                }
178.
179.                //else its some IO operation on some other socket
180.                for (i = 0; i < max_clients; i++) {
181.                    sd = client_socket[i];
182.                    if (FD_ISSET( sd , &readfds)) {
183.                        //Check if it was for closing , and also read the incoming messa
    ge
184.                        if ((valread = read( sd , buffer, 1024)) == 0)
185.                        {
186.                          //Stop the currently playing song (if there is one)
187.                          stop_song();
188.                          sleep(1);
189.
190.                          //Play a goodbye message
191.                          system("screen -dm bash -c \"omxplayer -
    o local /home/pi/seniordesign/Audio_Files/Speech/client_disconnected.mp3\"");
192.
193.                          //Make sure fan is off
194.                          digitalWrite (DESK_FAN_PIN, LOW);
195.
196.                          //Display message on LCD
197.                          lcd_client_disconnected(i);
198.
199.                          //Somebody disconnected , get his details and print
200.                          getpeername(sd , (struct sockaddr*)&address , \
201.                            (socklen_t*)&addrlen);
202.                          printf("Host disconnected , ip %s , port %d \n" ,
203.                              inet_ntoa(address.sin_addr) , ntohs(address.sin_port));

204.                          //Close the socket and mark as 0 in list for reuse
205.                          close( sd );
206.                          client_socket[i] = 0;
```

```
                        }

                        //Control GPIO pins
                        else
                        {
                            int data = atoi(buffer);
                            printf("\nreceived: %d\n", data);

                            control_hardware(data);

                            //Send back messages
                            //buffer[valread] = '\0';
                            //send(sd , buffer , strlen(buffer) , 0 );
                        }
                    }
                }
            }
        }
```

Main/hardware_controller.c
Contains functions which interact directly with hardware.

```
1.  //Shivam Kundan
2.  //S17-71-HOL1
3.  //Fall 2017
4.
5.  //Variable to keep track of how many on/off's for each device
6.  int fan_counter=0;
7.  int lamp_counter=0;
8.  int blender_counter=0;
9.  int song_counter=0;
10.
11. char *song_list[NUM_OF_SONGS]={"JingleBellRock.wav",
12.                       "mambo.wav",
13.                       "ThreeLittleBirds.mp3",
14.                       "GetLucky.mp3"};
15.
16. void lcd_client_connected(int i)
17. /*Display a message when a client connects.*/
18. {
19.     char *str_client = (char*)malloc(100);
20.     char i_str[10];
21.     strcpy(str_client,"python /home/pi/seniordesign/LCD_Scripts/lcd_client_connected.py
     ");
22.     sprintf(i_str, "%d", i+1);
23.     strcat(str_client,i_str);
24.     //strcat(str_client,"\"");
25.     system(str_client);
26. }
27.
28. void lcd_client_disconnected(int i)
29. /*Display a message when a client gets disconnected.*/
30. {
31.     char *str_client = (char*)malloc(100);
32.     char i_str[10];
33.     strcpy(str_client,"python /home/pi/seniordesign/LCD_Scripts/lcd_client_disconnected
    .py ");
34.     sprintf(i_str, "%d", i);
35.     strcat(str_client,i_str);
36.     //strcat(str_client,"\"");
```

```
37.     system(str_client);
38. }
39. 
40. void lcd_received_val(int i)
41. /*Print received value to LCD.
42.   This is helpful for a quick status check/debugging.*/
43. {
44.     char *str_client = (char*)malloc(100);
45.     char i_str[10];
46.     strcpy(str_client,"python /home/pi/seniordesign/LCD_Scripts/lcd_received_val.py ");
47.     sprintf(i_str, "%d", i);
48.     strcat(str_client,i_str);
49.     system(str_client);
50. }
51. 
52. 
53. void lcd_print(char line2[16])
54. /*Print to line 2 of LCD.*/
55. {
56.     char *str2 = (char*)malloc(100);
57.     strcpy(str2,"python /home/pi/seniordesign/LCD_Scripts/lcd_print.py ");
58.     strcat(str2,line2);
59.     system(str2);   //Execute in a detached terminal
60. }
61. 
62. void sig_handler(int signo)
63. /*Ensures proper termination of all running hardware/threads.*/
64. {
65.     //Turn off fan, lamp, and blender in case they are on
66.     system("gpio -g write 17 0");
67.     system("gpio -g write 27 0");
68.     system("gpio -g write 22 0");
69. 
70.     //Stop any music that is playing
71.     system("killall omxplayer.bin");
72.     sleep(1);
73. 
74.     //Print/speak messages
75.     printf("Bye!\n");
76.     system("screen -dm bash -c \"omxplayer -
    o local /home/pi/seniordesign/Audio_Files/Speech/system_off.mp3\"");
77.     system("python /home/pi/seniordesign/LCD_Scripts/lcd_exit.py");
78.     system("screen -dm bash -
    c \"python  /home/pi/seniordesign/LCD_Scripts/lcd_clear.py\"");
79. 
80.     //Kill all threads associated with this program
81.     system("killall screen") ;
82. 
83.     exit(0);
84. }
85. 
86. void clear_lcd()
87. /*Blanks out both lines of LCD.*/
88. {
89.     system("screen -dm bash -
    c \"python /home/pi/seniordesign/LCD_Scripts/clear_lcd.py\"");
90. }
91. 
92. void play_song()
93. /*Plays a song from song_list.*/
```

```
94. {
95.     printf("Playing %s\n",song_list[song_counter]); //Print name of song
96.     char *str1 = (char*)malloc(100);
97.     strcpy(str1,"screen -dm bash -c \"omxplayer -
    o local /home/pi/seniordesign/Audio_Files/Songs/");
98.     strcat(str1,song_list[song_counter]);
99.     strcat(str1,"\"");
100.            system(str1);   //Execute in a detached terminal
101.            if (song_counter!=NUM_OF_SONGS-1) song_counter++;
102.            else song_counter=0;
103.        }
104.
105.        void stop_song()
106.        /*Closes omxplayer.*/
107.        {
108.            system("killall omxplayer.bin");
109.        }
110.
111.        void measure_distance()
112.        /*Use ultrasonic sensor to measure distance.
113.          Use values to tell when door is open.
114.          For E127 lab, door closed is 150cm and
115.          door open is 152cm or more.*/
116.        {
117.            long unsigned  int start,stop,total,dist;
118.            struct timespec gettime_now;
119.            digitalWrite(outpin, LOW);
120.            sleep(0.01);
121.            digitalWrite(outpin, HIGH);
122.            sleep(0.01);
123.            digitalWrite(outpin, LOW);
124.
125.            //Trigger
126.            while (digitalRead(inpin)==LOW);
127.            clock_gettime(CLOCK_REALTIME,&gettime_now);
128.            start=gettime_now.tv_nsec;
129.            //printf("start: %ld\n",start);
130.
131.            //Echo
132.            while (digitalRead(inpin)==HIGH);
133.            clock_gettime(CLOCK_REALTIME,&gettime_now);
134.            stop=gettime_now.tv_nsec;
135.            //printf("stop: %ld\n",stop);
136.
137.            total=(stop-start);
138.            //printf("time elapsed: %lu\n",total);
139.            dist=(3.43*pow(10,-5)*total)/2;
140.            printf("distance: %lu cm\n",dist);
141.        }
142.
143.        void control_hardware(int data)
144.        /*Receives input integers and controls processes/threads accordingly.*/
145.        {
146.            //Fan
147.            if (data == DESK_FAN_ID) {
148.                fan_counter++;
149.                printf("fan counter: %d\n",fan_counter);
150.
151.                if (fan_counter%2==1) {
152.                  lcd_print("Fan On");
153.                  printf("Fan on\n");
```

```
154.                 digitalWrite (DESK_FAN_PIN, HIGH) ;
155.           }
156.               else {
157.                 printf("Fan off\n");
158.                 digitalWrite (DESK_FAN_PIN, LOW);
159.                 lcd_print("Fan Off");
160.           }
161.           }
162.           //Lamp
163.           else if (data == DESK_LAMP_ID) {
164.               lamp_counter++;
165.               printf("desk lamp counter: %d\n",lamp_counter);
166.               if (lamp_counter%2==1) {
167.                 lcd_print("Lamp on");
168.                 printf("Lamp on\n");
169.                 digitalWrite (DESK_LAMP_ID, HIGH);
170.           }
171.                 else {
172.                   lcd_print("Lamp off");
173.                   printf("Lamp off\n");
174.                   digitalWrite (DESK_LAMP_ID, LOW);
175.               }
176.           }
177.           //Blender
178.           else if (data == BLENDER_ID) {
179.               blender_counter++;
180.               lcd_print("Blender");
181.               printf("Blender\n");
182.
183.               if (blender_counter%2==1) {
184.                 lcd_print("Blender On");
185.                 printf("Blender on\n");
186.                 digitalWrite (BLENDER_PIN, HIGH) ;
187.           }
188.               else {
189.                 printf("Blender off\n");
190.                 digitalWrite (BLENDER_PIN, LOW);
191.                 lcd_print("Blender Off");
192.           }
193.           }
194.           //Play Song
195.           else if (data == PLAY_SONG) {
196.               lcd_print("Play Song");
197.               stop_song();
198.               printf("Play song\n");
199.               play_song();
200.           }
201.           //Next Song
202.           else if (data == NEXT_SONG) {
203.               lcd_print("Next Song");
204.               printf("Next song\n");
205.               stop_song();
206.               play_song();
207.           }
208.           //Stop Song (kill omxplayer)
209.           else if (data == STOP_SONG) {
210.               lcd_print("Stop Song");
211.               printf("Stop song\n");
212.               stop_song();
213.           }
214.           //Random Song
```

```
215.            else if (data == RANDOM_SONG) {
216.                lcd_print("Random song");
217.                printf("Random song\n");
218.                int random_num =rand()%NUM_OF_SONGS;
219.                char *str1 = (char*)malloc(100);
220.                strcpy(str1,"screen -dm bash -c \"omxplayer -
    o local /home/pi/seniordesign/Audio_Files/Songs/");
221.                strcat(str1,song_list[random_num]);
222.                strcat(str1,"\"");
223.                system(str1);   //Execute in a detached terminal
224.            }
225.            //Measure distance to door
226.            else if (data == ULTRASONIC_SENSOR_ID) {
227.                lcd_print("Door Sensor");
228.                printf("Ultrasonic distance sensor\n");
229.            //system("screen bash -c \"python sensor.py\"");
230.            //measure_distance();
231.            }
232.            //Temperature and Humidity
233.            else if (data == TEMP_SENSOR_ID) {
234.                lcd_print("Temp / Humidity");
235.                printf("Temperature and humidity\n");
236.                system("python /home/pi/seniordesign/Sensor_Scripts/temperature.py");
237.            }
238.            //Room brightness
239.            else if (data == LUX_SENSOR_ID) {
240.                lcd_print("Room Brightness");
241.                printf("Lux sensor\n");
242.                system("python /home/pi/seniordesign/Sensor_Scripts/light_sensor.py");
243.            }
244.            else {
245.              //Display unmapped value on LCD
246.              lcd_received_val(data);
247.              printf("Number not mapped to any functions\n");
248.            }
249.        }
```

Main/mappings.h

```
1.  //Shivam Kundan
2.  //S17-71-HOL1
3.  //Fall 2017
4.
5.  /*Values in braces are what the hololens client sends.*/
6.
7.  //Mapping BCM pins to wiringPi library
8.  #define DESK_FAN_PIN    0  //Desk Fan
9.  #define LAMP_PIN        2  //Desk Lamp
10. #define BLENDER_PIN     3  //Blender
11. #define inpin           5  //Trigger
12. #define outpin          4  //Echo
13.
14. //Hardware control mappings
15. #define DESK_FAN_ID         (1)
16. #define DESK_LAMP_ID        (2)
17. #define BLENDER_ID          (3)
18.
19. //Music Player mappings
20. #define PLAY_SONG           (4)
21. #define NEXT_SONG           (5)
22. #define STOP_SONG           (6)
23. #define RANDOM_SONG         (7)
24.
25. //Sensor control mappings
26. #define ULTRASONIC_SENSOR_ID (8)
27. #define TEMP_SENSOR_ID       (9)
28. #define LUX_SENSOR_ID        (10)
29.
30. //Others
31. #define NUM_OF_SONGS 4
```

Sensor_Scripts/clock.py

```
# Shivam Kundan
# S17-71-HOL1
# Fall 2017

# Constantly receive data from sensors, display on LCD, and save to file

# TSL2561 Light Sensor
# SI7021 Temperature & Humidity Sensor

import smbus
import time
import datetime
from subprocess import call
import sys
import shlex
import lcddriver

# Get I2C bus
bus = smbus.SMBus(1)

# Initialize LCD
display = lcddriver.lcd()

def brightness_sensor():
    # TSL2561 address, 0x39(57)
    # Select control register, 0x00(00) with command register, 0x80(128)
    #       0x03(03)    Power ON mode
    bus.write_byte_data(0x39, 0x00 | 0x80, 0x03)
    # TSL2561 address, 0x39(57)
    # Select timing register, 0x01(01) with command register, 0x80(128)
    #       0x02(02)    Nominal integration time = 402ms
    bus.write_byte_data(0x39, 0x01 | 0x80, 0x02)

    time.sleep(0.5)

    # Read data back from 0x0C(12) with command register, 0x80(128), 2 bytes
    # ch0 LSB, ch0 MSB
    data = bus.read_i2c_block_data(0x39, 0x0C | 0x80, 2)

    # Read data back from 0x0E(14) with command register, 0x80(128), 2 bytes
    # ch1 LSB, ch1 MSB
    data1 = bus.read_i2c_block_data(0x39, 0x0E | 0x80, 2)

    # Convert the data
    ch0 = data[1] * 256 + data[0]
    ch1 = data1[1] * 256 + data1[0]
    brightness = ch0-ch1

    # Output data to screen
    #print ("Full Spectrum(IR + Visible) :%d lux" %ch0)
    #print ("Infrared Value :%d lux" %ch1)
    #print ("Visible Value :%d lux" %brightness)
    return brightness

def humidity_sensor():
    # SI7021 address, 0x40(64)
    #       0xF5(245)   Select Relative Humidity NO HOLD master mode
    bus.write_byte(0x40, 0xF5)
```

```
59.
60.     time.sleep(0.3)
61.
62.     # SI7021 address, 0x40(64)
63.     # Read data back, 2 bytes, Humidity MSB first
64.     data0 = bus.read_byte(0x40)
65.     data1 = bus.read_byte(0x40)
66.
67.     # Convert the data
68.     humidity = float(((data0 * 256 + data1) * 125 / 65536.0) - 6)
69.     lcdHumidity=str(round(humidity)).split('.')[0]
70.     a,b=str(humidity).split('.')
71.     logHumidity=float('.'.join((a, b[0:3])))
72.     #print ("Relative Humidity: %.2f %%" %logHumidity)
73.     return (lcdHumidity,logHumidity)
74.
75. def temperature_sensor():
76.     time.sleep(0.3)
77.     # SI7021 address, 0x40(64)
78.     #       0xF3(243)   Select temperature NO HOLD master mode
79.     bus.write_byte(0x40, 0xF3)
80.
81.     time.sleep(0.3)
82.
83.     # SI7021 address, 0x40(64)
84.     # Read data back, 2 bytes, Temperature MSB first
85.     data0 = bus.read_byte(0x40)
86.     data1 = bus.read_byte(0x40)
87.
88.     # Convert the data
89.     cTemp = ((data0 * 256 + data1) * 175.72 / 65536.0) - 46.85
90.     a,b=str(cTemp).split('.')
91.     lcdTemp=float('.'.join((a, b[0:1])))
92.     logTemp=float('.'.join((a, b[0:3])))
93.     fTemp = cTemp * 1.8 + 32
94.
95.     # Output data to screen
96.     #print ("Temperature in Celsius is : %.2f C" %cTemp)
97.     #print ("Temperature in Fahrenheit is : %.2f F" %fTemp)
98.
99.     return (lcdTemp,logTemp)
100.
101.        def get_date_time():
102.            now = datetime.datetime.now()
103.            myDate=now.strftime('%a,%b%-d')     #Short value because of 16-char lcd
104.            myTime=now.strftime('%-I:%M%p')
105.            logDate=now.strftime('%m/%d/%y')    #The logged date is in a different forma
    t than lcd's date
106.            logTime=now.strftime('%-I:%M:%S%p')
107.            return (myDate,myTime,logDate,logTime)
108.
109.        try:
110.            display.lcd_display_string("                ", 1)
111.            display.lcd_display_string("                ", 2)
112.            num_writes=0
113.            flag=0
114.            b_s_last=0
115.            while True:
116.                if (flag!=0):
117.                            b_s_last=b_s
118.
```

```
119.                #Get current values
120.                myDate,myTime,logDate,logTime=get_date_time()
121.                lcdTemp,logTemp=temperature_sensor()
122.                lcdHumidity,logHumidity=humidity_sensor()
123.                b_s=brightness_sensor()
124.                flag=1
125.                print(b_s_last)
126.                print(b_s)
127.
128.                #Automated night light 'smart service'
129.                if (abs(b_s-b_s_last)>300):
130.                                print("("+myDate+" "+str(logTime)+")"+"\n"+str(logTemp)+"C "
    +str(logHumidity)+"% "+str(b_s)+"lux "+"\n")
131.                                if (b_s<100):
132.                                    call(["omxplayer","-
    o","local","/home/pi/seniordesign/Audio_Files/Speech/main_lights_off.mp3"])
133.                                    call(["gpio","-g", "write", "27", "1"])
134.                                elif (b_s>100 and b_s<500):
135.                                    call(["omxplayer","-
    o","local","/home/pi/seniordesign/Audio_Files/Speech/dim_lights.mp3"])
136.                                    call(["gpio","-g", "write", "27", "0"])
137.                                if (b_s>500):
138.                                    call(["omxplayer","-
    o","local","/home/pi/seniordesign/Audio_Files/Speech/main_lights_on.mp3"])
139.                                    call(["gpio","-g", "write", "27", "0"])
140.
141.                #Print to lcd
142.                l1=myDate+" "+myTime
143.                display.lcd_display_string(l1, 1)
144.                l2=str(lcdTemp)+"C "+str(lcdHumidity)+"% "+str(b_s)+"Lux"
145.                display.lcd_display_string("                ", 2)
146.                display.lcd_display_string(l2, 2)
147.
148.                #Write to log file
149.                f=open('lab_sensors_log.csv','a')
150.                outstring = logDate+","+str(logTime)+','+str(logTemp)+"C"+','+str(logHum
    idity)+"%"+','+str(b_s)+"lux"+"\n"
151.                f.write(outstring)
152.                num_writes+=1
153.                f.close()
154.                print("#writes: " + str(num_writes))
155.
156.                time.sleep(2) #Makes a total of 4 reading/minute
157.
158.        except KeyboardInterrupt:
159.            #print("Cleaning up!")
160.            display.lcd_clear()
161.            s="#writes: " + str(num_writes)
162.            display.lcd_display_string(s, 2)
```

9.7 Appendix J: Resumes

Shivam Kundan

shivamkundan@siu.edu • (217) 974 5324 • github.com/shivamkundan
710 S Illinois Ave, Carbondale, IL – 62901

Professional Summary

- Member of ATMAE robotics chapter for Southern Illinois University.
- Achieved Dean's list status every semester while working 16-20 hours per week.

Education

Bachelor of Science in Computer Engineering — Southern Illinois University, Carbondale, IL
Graduation: December 2017 — **GPA: 3.65/4.0**

Relevant Coursework

- Digital Circuit Design w/ HDL
- Digital VLSI Design
- Data Structures in C, C++, Python
- Linux Systems and Parallel Programming
- Computer Systems Architecture
- Controllers: Raspberry Pi, Arduino
- Matlab
- Principles of Operating Systems

Experience

Internships

- (June 2014 – August 2014) **Student Trainee at Honeywell Automation India Limited**: Using Honeywell's distributed control systems, programmable logic controllers, and control and data acquisition systems in their client's chemical manufacturing plant.

- (June 2012 – August 2012) **Software Development Intern at Unique Identification Authority of India**: Wrote a program using Java, MySQL, and MS Excel to streamline the process of reviewing scholarship applications for new college students.

Student Jobs

- (November 2015 – March 2017) **Technical Assistant for the IT Department at College of Education, SIU:** Troubleshooting Windows and Mac computers, network configuration, hardware and software support. Job also involved handling large amounts of sensitive data and writing weekly newsletters.

Technical Projects

Microsoft HoloLens: Object Recognition and Internet of Things (IoT)

- Augmented-reality application for the Microsoft HoloLens.
- Using object recognition and hand gestures to interact with real world objects such as sensors, lights, electronics, etc. Aim is to convert a room to a 'smart room'.
- Software Tools: Unity 3D Game Engine, Vuforia Augmented Reality, HoloLens Emulator.
- Hardware: Raspberry Pi and Microsoft HoloLens.

Robotics: Autonomous and Manual Navigation

- Worked in a team to design the control methodology for autonomous as well as manual navigation of robot through an obstacle course.
- Awarded **2nd prize** at 2015 ATMAE Robotics Competition in Pittsburgh, PA.
- Implemented design using Python code on two communicating Raspberry Pi's.

Systems Programming: Simple Operating System

- Implemented a shell using C and POSIX API.
- Can execute, terminate and display currently running processes.
- Job scheduling is handled using a Round Robin method.

Computer Architecture: MIPS CPU Implementation

- 32-bit MIPS datapath, memory, control, and storage written in Verilog.
- Supports 8 arithmetic instructions, 2 load/save instructions, and 4 branch instructions.

Key Skills

- **Languages:** C, C++, MIPS Assembly, Python, Matlab, SQL
- **Hardware Description Languages:** Verilog, VHDL
- **Systems Programming:** UNIX kernel development, scheduling, memory control, multi-threading, GDB
- **Electronic Design Automation (VLSI design):** Atalanta-M Automatic Test Pattern Generator
- Star Trek trivia.

1305 Point Drive, Carbondale,
Illinois 62901

Phone (618)303-7469
E-mail: isaakonoh@gmail.com

Isaac Onoh

Objective

To be part of a dynamic electrical or computer engineering team, where I can utilize and contribute my strong analytical skills and technical background while gaining valuable experience and building stronger team working skills.

Education

Southern Illinois University (SIU), Carbondale, IL **Dec 2017**
Bachelor of Science, Electrical and Computer Engineering;
Minor in Mathematics **GPA: 3.17**

Experience

Research, Mixed Reality Developer, SIU, Carbondale, IL **Jan 2017 – Dec 2017**

- Currently leading the development of a mixed reality environment for my senior design project.
- Developed capability in the utilization of Microsoft's HoloLens and Unity Development tool needed to support Holographic computing.
- Demonstrated leadership through organization of the research work, consulting with Supervisor, managing my teammates and assigning jobs.

Internship, Research Assistant, SIU, Carbondale, IL **May 2017 – August 2017**

- Developed capability in utilizing Microsoft's Azure services regarding the internet of things.
- Gained understanding of how the field of the internet of things works.
- Gained valuable experience working with a raspberry pi computing device as well as with shell scripting on the Raspbian operating system.

Internship, Technical Support, Chevron, Nigeria **May 2013 – July 2013**

- Leveraged computer engineering skills to replace non-functional system units.
- Effectively participated in the deployment of company-wide operating system migration from GIL3 to GIL3.5.
- Effectively on-boarded, using computing tools, assisted new employees in managing their request for Identity/access control cards that authorized them to use company premises and resources.

Technical Support, Saluki-Tech, SIU, Carbondale, IL **Aug 2014 – Dec 2014**

- Re-installed operating systems for tablets and faulty laptops.
- Leveraged problem-solving skills to resolve software issues with regards to Wi-Fi connectivity for students.
- Consistently provided Help Desk support for students by resolving students computing issues through use of phone and effective interpersonal relationship skills.
- Setup email accounts on portable devices, as well as performed password resets to student accounts.

Computing Skills

MATLAB, C++, Verilog, Unity Development Tool, Microsoft Office (Word, Excel, PowerPoint), Windows Operating system, Linux Operating system.

Awards

Dean's List: Fall 2012, Fall 2016
SIUC books scholarship – Spring 2013
Chevron REACH scholarship - 2012

Hobbies

Soccer, Table tennis (Ping-Pong), and Musical Instruments (piano, saxophone, bass guitar)

Antonio N. Pugh

1207 S Wall St.
Carbondale, IL 62901
317-694-9434
apugh19@siu.edu

Objective: To pursue an internship that would allow personal and professional growth in the area of computer engineering and computer science.

Education

2009-2013 Pike High School, 6701 Zionsville Rd., Indianapolis, Indiana
All Honors Classes

2013-Present Southern Illinois University,1263 Lincoln Dr.,Carbondale Illinois
Major: Computer Engineering, Minor: Computer Science & Math.

Projects

2017 Microsoft Hololens IoT Smart Room
Responsibilities: Hardware Specialist
(https://youtu.be/K9n25WHqLxM)

Work Experience

2014-2015 SIU Concessions, Southern Illinois University, 1263 Lincoln Dr, Carbondale, IL
Responsibilities: Concessions.

2015 Pearson Education, 5550 W. 74th St, Indianapolis IN
Responsibilities: Material Handler

2015-2016 SIU Housing, Southern Illinois University, 1263 Lincoln Dr, Carbondale, IL
Responsibilities: Resident Assistant

2016 TCC Software Solutions, 1022 East 52nd Street, Indianapolis, IN
Responsibilities: Software Consultant

Extra Curricular

2009 to 2011 ACE (Architectural, Civil and Electrical Engineering) Mentor Program at Pike High School

2006 to 2013 MEPI (Minority Engineering Program of Indianapolis)

2011 CLD (Center for Leadership Development)

2011 to 2013 NSBE (National Society of Black Engineers)

KEN NAKAZAWA

905 E Park Street Apt. E4• Carbondale, IL 62901 • (618) 305-3236 • ken.nakazawa@siu.edu

EDUCATION

Bachelor of Science in Computer Engineering — Expected Graduation Date: December, 2017
Southern Illinois University Carbondale, Carbondale, IL — GPA of the major: 3.77
Dean's List-2015 to 2017 Fall

Certificate of English for Academic Purposes — Graduation Date: March 2013
Center for English as a Second Language,
Southern Illinois University Carbondale, IL

High School Diploma
Kofu Higashi High School -Science and Math Course- — Graduation Date: March 2012
Course focused on high level of mathematics and science- — GPA: 3.29
Yamanashi-Kenritsu Kofu Higashi High School, Yamanashi, Japan

LANGUAGES

- English: Fluent
- Japanese: Fluent

RELATED COURSEWORK

- Programming using C Language
- Programming using C++,C# Language
- Programming using Python Language
- Programming using Java Language
- Programming using Ruby Language
- Design and Evaluation of Digital Circuit
- Design and Evaluation of Computer Aided Digital Circuit
- Design and Evaluation of Computer hardware
- Design and Evaluation using Verilog & VHDL
- Operation of different operating systems include Windows, MAC OS X, Linux, Unix and Raspbian.
- Signal Processing and telecommunication
- Developing Algorithm and Software
- Design and evaluation of 3D VR using Unity

EXPERIENCE

Finance, Japanese Student Association, RSO, Southern Illinois University, IL
September2015- present
- Developed a system that integrates both financial movement and activity schedules.

President, Japanese Student Association, RSO, Southern Illinois University, IL
September2013 - September 2015
- Managed the association that has many Japanese students and communicated with other countries' presidents or students in order to succeed during the international fair.
- Developed communication ability in order to complete a project with people from different countries.
- Observed and learned how people and organization move during the complicated and long projects.

Tutor, Department of Foreign Language, Southern Illinois University, IL
September 2013 – December 2013
- Tutored Japanese to adult learners of diverse nationalities and backgrounds who needed additional help while taking Japanese classes.

SKILLS

- Programming using several languages (C, C++, C#, Java, Python, HTML,and Ruby)
- Designing and Evaluation of digital, electronic, and integrated circuit
- Designing Hardware and Software
- Cloud Computing and usage of big-data using Hadoop
- Engineering Microwave, photon, electron, antenna systems, fiber optic systems.
- Developing VLSI, embedded systems, microcontrollers, multicore architectures, and networks
- Operation of VCSMX, Sim Vision, Encounter RTL Compiler, Design Compiler, Symphony C Compiler.
- Design and Build a computer from the parts
- Operation of different Operating System(Windows, MAC OS X, Linux, Unix, and Raspbian)
- Computer Literacy: Wiki, Blog, Microsoft (Word, Excel, Outlook, Power Point),Microsoft HoloLens
- Technological Tools: QR codes, Vialogues, Visual Studio, Google Drive, Photoshop, Weebly, Matlab, Unity
- Speaking Japanese and English Fluently