ECE 53301: Wireless and Multimedia Computing

Final Project Report – Group 1

Inventory Automation Using Electronically Connected Intelligent Shelves

Priyank Kalgaonkar	-	Team Leader
Sahil Kumar	-	Team Member
Linknath Surya Balasubramanian	-	Team Member

Department of Electrical and Computer Engineering,
Purdue School of Engineering and Technology at IUPUI.

Submitted to:

Course Instructor
Dr. Mohamed El-Sharkawy
Department of Electrical and Computer Engineering, IUPUI.

Submitted as partial fulfillment for the requirement of Fall 2019 - ECE 53301-26877: Wireless and Multimedia Computing course.

Date of Submission
December 09, 2019

Executive Summary

The Electronically Connected Intelligent Shelves (ECIS) system uses a microcontroller unit, ultrasonic sensors and IoT server. The main objective of the Inventory Automation Using Electronically Connected Intelligent Shelves project is to reduce loss in revenue due to delayed shelf-restocking (when product is in-stock in the store but not stocked on the shelf) and inaccurate forecasting (under-estimating future product sales) practices. The result is to demonstrate this idea of inventory automation using ECIS system by enabling the means to monitor and track store inventory in real-time, perform data analysis remotely in cloud, improve shopping experience for the consumers and increase revenue for the retailers in the retail industry.

A study conducted by the Grocery Manufacturers of America, which surveyed 71,000 consumers in 661 retail outlets, found that the average out-of-stock rate [1][2] in a traditional consumer retail store is 7.9% and it costs retailers 4% loss in category sales. Amazon.com Inc., a multi-national e-commerce and technology company based in Seattle, Washington, is currently pilot testing a similar idea of electronic shelves using weight sensors as well as QR codes, video cameras and computer vision systems in its Amazon Go physical stores to increase efficiency throughput, reduce labor-costs, increase revenue and improve overall customer shopping experience.[3]

Our design of ECIS system prototype includes an array of ultrasonic sensors, which can be retrofitted in existing shelves with minimal modifications or built-in in to new shelves, connected wirelessly to a central server from where the inventory of goods on the shelf can be monitored in real-time as well as data acquired from these sensors can be used to perform predictive analysis using data mining, feature engineering and machine learning techniques to better predict future product sales and minimize inaccurate forecasting instances as aforementioned.

This report provides detailed information that will be useful for the retailers interested in this concept and the engineering team(s) responsible for further research and development.

[1] The out-of-stock rate is the percentage of SKUs that are out-of-stock in a retail store at a given point of time.

Acknowledgement

This project consumed an immense amount of work, research, and dedication. Nonetheless, implementation would not have been possible without the support of many individuals and organizations.

We are highly indebted to our professor: Dr. Mohamed El-Sharkawy and our teaching assistants: Dewant Katare and Niranjan Ravi at the Purdue School of Engineering and Technology at IUPUI for their guidance and support in completing this project. During the course of this work, their constant association with my team has been most pleasurable. Therefore, on behalf of all my team members, I would like to extend our sincere gratitude to all of them.

We are also grateful to Mr. Jeffrey Sears who provided us with access to the SL111 lab in the Engineering Science & Technology (SL) Building and with necessary tools to build, develop, and test hardware components, and subsystems for this project. Without his time and support in providing necessary tools and components, our hardware team would have been greatly challenged to develop, integrate and test the hardware subsystems and components for this project.

My thanks and appreciations also go to my colleagues involved in developing this project and specially to those who have willingly helped me out with abilities at their best. I would like to extend my special gratitude to my team members: Sahil Kumar and Linknath Surya Balasubramanian, the members of my team actively involved in this project.

We would like to express our sincere thanks towards volunteer helpers, family, and friends who provided us with essential support and additional resources in the implementation of this project.

Without the help and counsel of the aforementioned individuals and organizations, always generously and unstintingly given, the completion of this work would have been immensely difficult and time consuming.

Priyank Kalgaonkar,
Team Leader.

Table of Contents

List of Tables

Table

List of Figures

Figure

Glossary

A

AP
Access Point: A networking hardware device that allows other Wi-Fi devices to connect to a wired network.

B

C

Cloud
Data centers and services available to many users over the Internet.

D

E

ECIS System
Electronically Connected Intelligent Shelves.

F

G

H

I

Isle
A passageway for people to walk through in a grocery store.

J

K

L

M

MCU
Micro Controller Unit: A computer on a single chip.

Module
Any of a number of distinct but interrelated units from which a program may be built up or into which a complex activity may be analyzed.

MVC
An application design model comprised of three interconnected parts. They include the model (data), the view (user interface), and the controller (processes that handle input).

N

O

P

Q

R

ROI
Return on Investment (ROI) is a ratio between net profit and cost of investment.

S

T

U

Ultrasonic
Involving sound waves with a frequency above the upper limit of human hearing.

V

W

X

Wi-Fi
A facility allowing computers, smartphones, or other devices to connect to the Internet or communicate with one another wirelessly within a particular area.

Y

Z

1.0 INTRODUCTION

The traditional shelf restocking (replenishment) technique, where an employee at the store makes physical rounds after a pre-determined interval of time (in hours) and then informs the on-site warehouse of out-of-stock goods on the shelf, results in loss in revenue of that particular consumer good not being sold even though the product is in-stock in the on-site warehouse. Delays and/or failures in internal processes and constraint human resources ultimately results in dissatisfaction of customers due to ‘out-of-stock’ goods in the store and can possibly result in total abandonment of shopping cart and loss of customer’s business. This is one of the biggest challenges the physical retail industry is currently facing.

The idea of the Electronically Connected Intelligent Shelves (ECIS) system relates to an array of sensors, each placed behind each row of products (consumer goods), centrally connected to a microprocessor unit, which then processes the data and sends it to cloud for alerting and data analysis purposes. Most modern shelves at a retail store are either made of metal or a combination of metal and plastic and are designed to nest into one another by drilling a two to three holes. This facilitates easy and quick installation of the sensors and system, and ultimately aids in shopping experience for consumers and increased revenue for retailers over the long run.

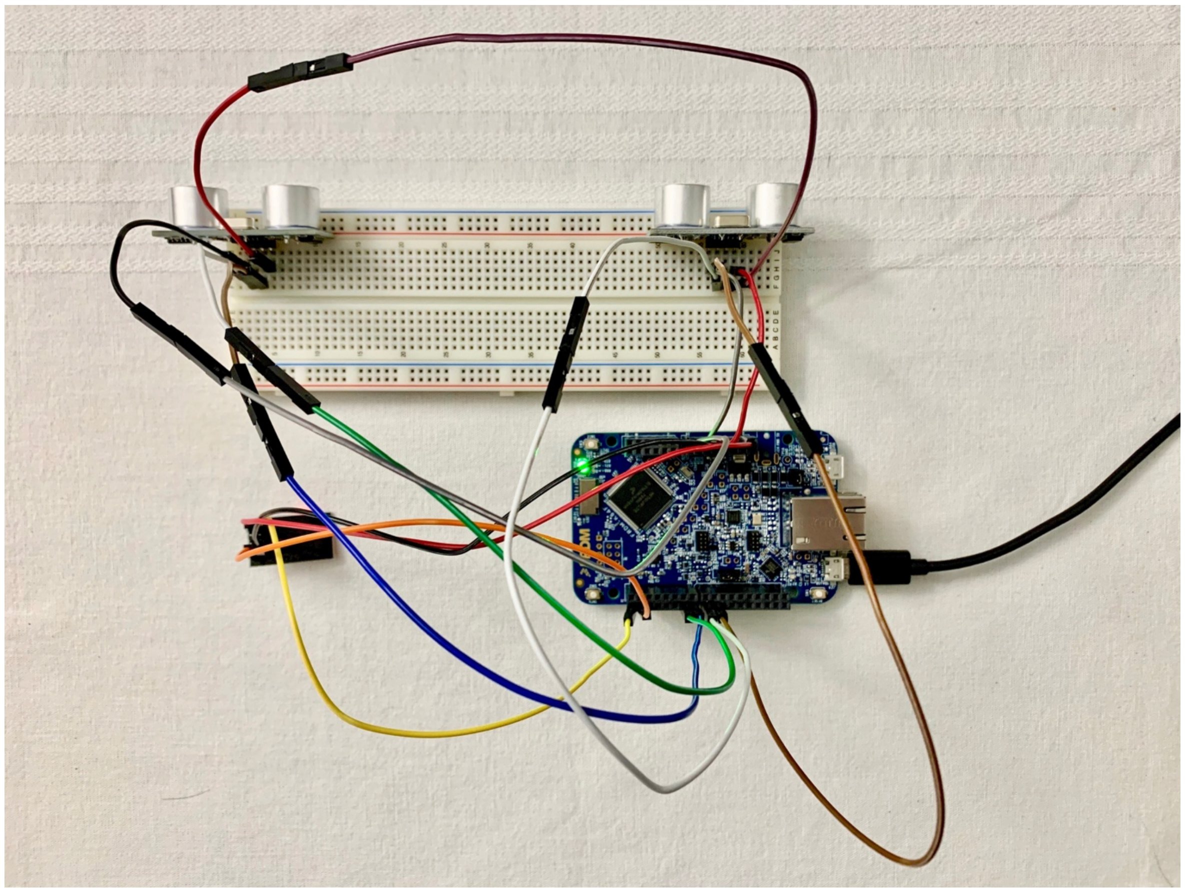


Figure 1.1 Picture of ECIS System Prototype

The design, results, and expected impact of the ECIS system will be apparent in the detailed sections below.

2.0 PROBLEM STATEMENT

2.1 Need

As stated in the introduction, the inefficient traditional process of out-of-stock product replenishment results in loss in revenue and dissatisfaction of customers due to ‘out-of-stock’ goods in the store. A less time-consuming process that does not rely on store employees checking every item in every isle of the store is needed to minimize the loss in revenue due to delayed shelf-restocking (when product is in-stock in the store but not stocked on the shelf) and inaccurate forecasting (under-estimating future product sales) practices.

2.2 Objective

To reduce the strain and increase efficiency, we are introducing the idea of inventory automation using electronically connected intelligent shelves. By allowing the on-site warehouse to monitor shelves in real-time and replenish depleting shelves when necessary, it will help the overall system to be more efficient which will ultimately help, both, the consumers and the retail business. This idea of ECIS system also opens door to predictive analysis using machine learning and data mining techniques, and for researching shopping trend of a particular good or category.

2.3 Background and Technology Survey

Amazon.com Inc., a multi-national e-commerce and technology company based in Seattle, Washington, is currently pilot testing a similar idea of electronic shelves using weight sensors as well as QR codes, video cameras and computer vision systems in its Amazon Go physical stores to increase efficiency throughput, reduce labor-costs, increase revenue and improve overall customer shopping experience.[3]

Another design of electronic shelves is the Smart Shelf by PointOfSale.com. In this system, weighing scales have been installed into the shelves. These weight sensors consistently notify the back-end system about the existing quantity of items on the shelves and sends an alert to the warehouse when the shelf is empty.[4]

2.4 Marketing Requirements

The marketing requirements are focused on making the ECIS system cost effective and easy to implement in a retail environment. They are as follows:

1. Data should be provided by the system in real-time.
2. Few modifications should be required for installation in current traditional shelves.
3. The system should be affordable for retailers to purchase and maintain.
4. The system should work with existing products.
5. The system should provide enough data for data analytics.

The detailed engineering requirements derived from these marketing requirements are listed in section 3.3.

3.0 SYSTEM ARCHITECTURE

3.1 System Architecture

The overall architecture of the ECIS system has at its center the NXP FRDM-K64F MCU featuring an ARM Cortex M4 core running up to 120MHz and embedding 1024KB Flash, 256KB RAM. The FRDM-K64F MCU receives data from HC-SR04 ultrasonic sensor(s). This sensor module includes ultrasonic transmitters, receiver and control circuit. In an auxiliary role is the Wi-Fi Signal Transmitter/Receiver: ESP8266 module, which the FRDM-K64F MCU uses to communicate with the IoT server: ThingSpeak. The functional architecture is drawn in Figure 3.1.1 as follows:

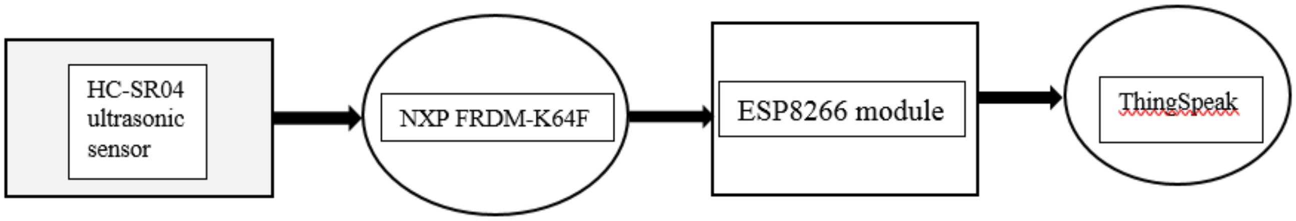


Figure 3.1.1: System Architecture

3.2 Activity Diagram

As pictured in Figure 3.2.1 below, the process begins when the ultrasonic sensor measures a change in distance (in centimeters). This data is then sent to FRDM-K64F MCU which calculates percentage of the remaining product and sent to the central cloud server as well as to the Warehouse Central Monitoring Unit (WCMU). If this calculated value is beyond pre-defined threshold for out-stock reading, an alert is displayed on the screen of the WCMU and red LED is displayed on FRDM-K64F MCU alerting the store associates to replenish that particular shelf. These ultrasonic sensors consistently monitor and notify the WCMU about the existing quantity of products on the shelves.

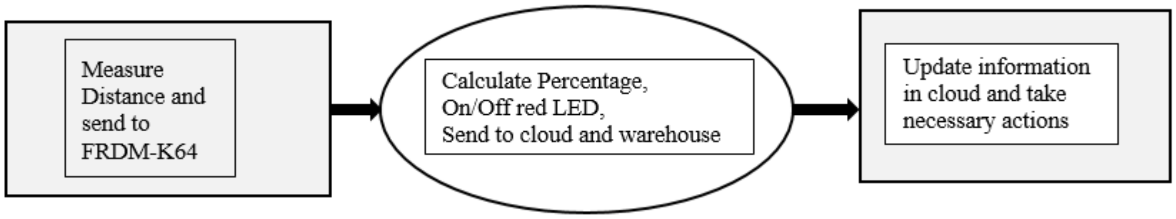

Measure Distance and send to FRDM-K64
Calculate Percentage, On/Off red LED, Send to cloud and warehouse
Update information in cloud and take necessary actions

Figure 3.2.1: Activity Diagram for the ECIS System.

3.3 Engineering Requirements

To satisfy the marketing requirements explained in section 2.4, we have set the engineering requirements listed below:

Marketing Require-ment	Engineering Requirement	Justification
1	System should process data in real-time.	Ultrasonic sensors communicate with FRDM-K64F (MCU) consistently per the design. MCU then pushes data to cloud and WMCU.
2	System should be inexpensive to implement and deploy.	Ultrasonic sensors and MCU cost a fraction when compared to weight/motion sensors, and camera and motion tracking system. Ultrasonic sensors require two small screws to be mounted on the back side of the shelves. Wires to MCU can then be run along the edges of the shelf. MCU connects with local AP wirelessly.
3	System should work with existing products.	Ultrasonic sensors can work in the dark and detect all solid objects.
4	System should provide data for analysis and research purposes.	This system communicates with cloud server, ThingSpeak, where data over time from the sensors is populated which can be used for data analysis, estimations, and related studies and research.

Table 3.3.1: Engineering Requirements Justification Table

3.4 Validation

The following analyses explain how our design is expected to meet the engineering requirements. Each explanation is listed under the corresponding engineering requirement, reprinted below for convenience.

1. **System should process data in real-time.**
   This requirement is met by the NXP FRDM-K64F MCU we are using. This MCU features an ARM Cortex M4 core running up to 120MHz, which is enough to process data received from the HC-SR04 ultrasound sensor module(s).

2. **System should be inexpensive to implement and deploy.**

   Multiple ultrasonic sensors (an array) can be connected to a single NXP FRDM-K64F MCU. In addition, these sensors are very inexpensive and cost of parts to install is minimal as well. Ultrasonic sensors require two small screws to be mounted on the back side of the shelves. Wires to MCU can then be run along the edges of the shelf. MCU connects with local AP wirelessly. Hence, the cost of the entire system stays low.

3. **System should work with existing products.**

   Ultrasonic sensors can work in the dark and detect all solid objects. This module provides 2cm - 400cm non-contact measurement function. The ranging accuracy can reach up to 3mm, which is actually much accurate than we need.

4. **System should provide data for analysis and research purposes.**

   This prototype is configured to communicated with the cloud server: ThingSpeak, wirelessly. Data obtained from the ultrasound sensors will be sent to the cloud server instantly, either continuously or at certain pre-determined intervals of time, which can then be populated and used for further analysis and studies. Since this data is stored in cloud, it can be accessed and analyzed by any authorized personnel around the world.

4.0 STANDARDS COMPLIANCE

4.1 Hardware Standards

All the components used in this system are bought off-shelf from the electronics store, which are manufactured in compliance with the following standards and hence, the overall system, when fully assembled, will be in compliance with the standards described below in detail:

A. **Safety and Health Compliance Standards:**

EN 60950-1:2006 + A1:2010 + A11:2009 + A12:2011: The safety standard EN 60950-1 applies to mains and battery powered equipment such as the NXP FRDM-K64F, HC-SR04 Ultrasonic Sensor, and ESP8266 Wi-Fi Module. "Requirements within this standard address normal equipment operating conditions, likely and consequential faults, foreseeable misuse and external influences such as temperature, altitude, pollution, moisture and over voltages. In general, these requirements are intended to minimize the risk of fire, electric shock and injury for the operator, layman, service person and anyone else that may come into contact with the equipment during installation, operation and maintenance." [5][6][7]

B. **Electromagnetic Compatibility (EMC) Compliance Standards:**

EN 55022:2010: The safety standard EN 55022 applies to the Radio disturbance characteristics of the components in the system. All the components of the system have been tested by the manufacturer and found to comply with the limits for Class B Information Technology Equipment according to the European Standard except the touch screen display which complied with the limits for Class A Information Technology Equipment according to the European Standard. [5][6][7]

C. **Federal Communications Commission (FCC) Compliance Standards:**

FCC Emissions Compliance Statement: All the components of the system have been tested by their respective manufacturers and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. All the components of the system comply with Part 15 of the FCC Rules. Operation is subject to the following two conditions:

a) This system might not cause harmful interference

b) This system must accept any interference received, including interference that might cause undesired operation. [5][6][7]

D. Restriction of Hazardous Substances Directive (RoHS) Compliance Standards:

RoHS Directive Compliance Statement: NXP FRDM-K64F MCU board and other components of the ECIS system complies with the relevant provisions of the RoHS Directive for the European Union. In common with all Electrical and Electronic Equipment (EEE), the NXP FRDM-K64F MCU or the any components of the ECIS system should not be disposed of as a household waste. Alternative arrangements may apply in other jurisdictions. [5][6][7]

4.2 Software Standards

Since our project includes custom software developed in C programming language in Mbed Software Development Kit (SDK), it is expected to adhere to Mbed SDK's standards. The SDK is licensed under the permissive Apache 2.0 (Open-Source) license and so it can be used in both commercial and personal projects. The software was developed by us, and existing libraries were imported directly into our complier from Mbed's library repositories built on top of the SDK by the Mbed Developer Community.[8]

5.0 DETAILED DESIGN

5.1 Hardware Design

Since our prototype is designed to be retrofitted on to existing shelves with minimal modification or to be incorporated in the design of new shelves, we have setup a prototype of our ECIS system by utilizing a simple breadboard and connecting wires as shown below:

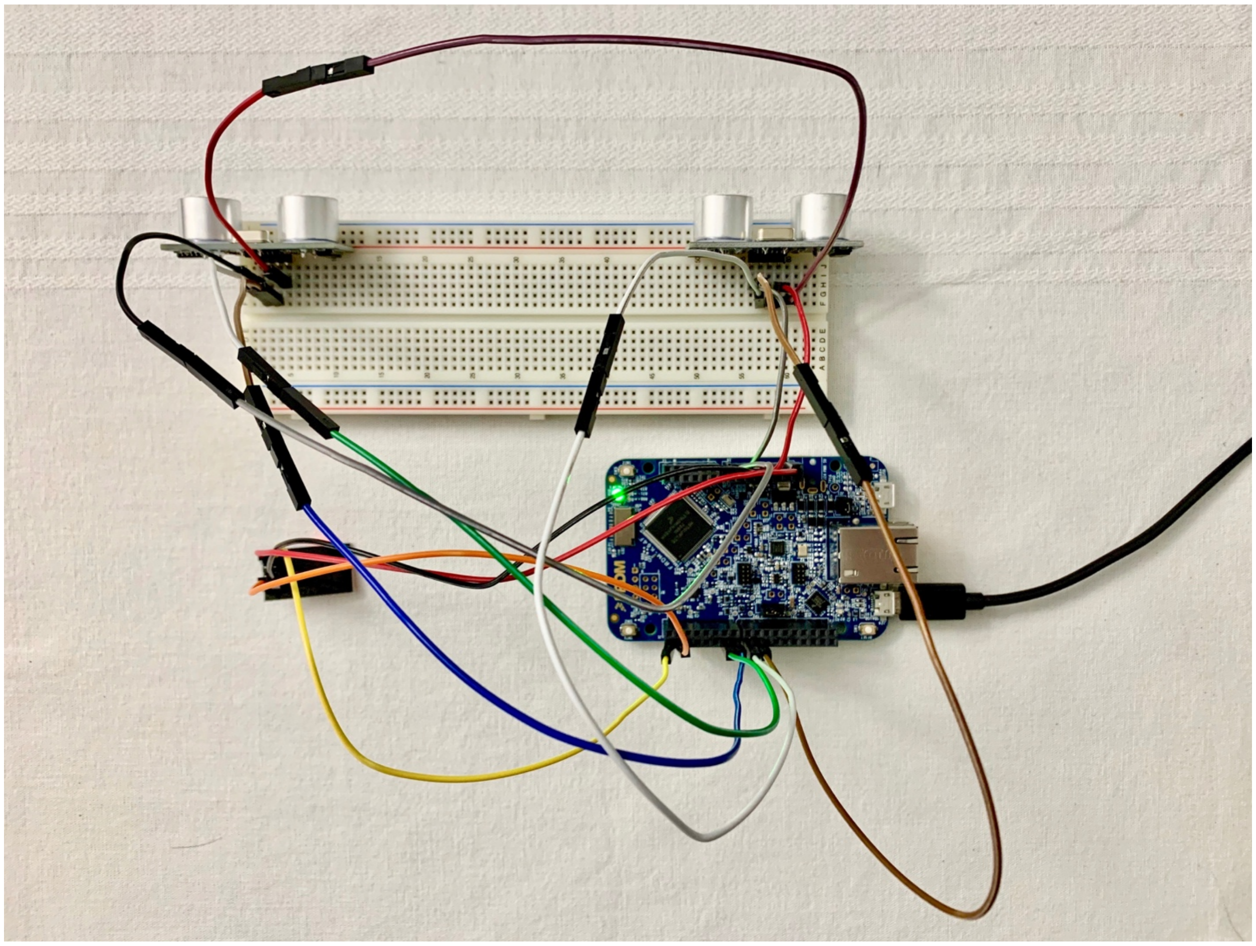


Figure 5.1.1: ECIS System Prototype

5.1.1 HC-SR04 Ultrasonic Sensor Module:

The modules ultrasonic sensing module consists of one of each: ultrasonic transmitter, receiver and control circuit. Ultrasound is a high-pitched sound of frequencies greater than human's audible limit of hearing; ideal range: 20 Hz to 20 kHz [9]. It is configured to work as follows:

1. A pulse of 10µS+ is applied to the Trigger (TRIG) pin. The sensor then transmits a sonic burst of eight rapid pulses at 40 KHz. Now, the ECHO pin goes HIGH.

a. Case 1: If the reflected pulses are not detected by the sensor's receiver, for example: after 38ms, ECHO pin will time out after 38ms and return LOW.

b. Case 2: If the reflected pulses are detected by the sensor's receiver, the ECHO pin immediately returns LOW. This indicates that an object was detected. The distance is then calculated by the following formula:

$$Distance\ (in\ cm) = \frac{[\left(0.034\ \frac{cm}{\mu s}\right) * 500\mu s]}{2}$$

where, speed of sound is 340 m/s.

An example of the code to calculate the distance using the formula above is as follows:

```
void HCSR04::start(void)
{
    trigger = 1;
    wait_us(10);
    trigger = 0;
}

void HCSR04::isr_fall(void)
{
    pulsetime.stop();
    pulsedur = pulsetime.read_us();
    distance = (pulsedur*343)/20000;
    pulsetime.reset();
}
```

5.2 Component Dimension

Measurements of the main components (excluding wires) are shown in Table 5.1.2 below:

Component	Height (mm)	Width (mm)	Depth (mm)
NXP FRDM-K64F MCU	81	53	7
HC-SR04 Ultrasonic Sensor	30	40	8
ESP8266 Wi-Fi Module	10	23	3

Table 5.2.1 Table for Component Dimensions.

5.3 Software Design

The software was developed in Mbed Software Development Kit (SDK) utilizing C Programming Language, which is one of the recommended programming languages. The software design consists of:

1. Starting up the code.
2. Checking C Library integration.
3. Checking Peripheral libraries.
4. Executing the code.

MVC stands for Model, View, and Controller — the three main elements of an MVC design as follows:

- Model: The model is the representation of the data. In this case, the model is readings received from the ultrasonic sensor.
- View: The view is the user-interface portion of the application; it describes where the data will be displayed on the screen. In this case, View is Tera Term.
- Controller: The controller coordinates the model and the view. In this case, Controller is FRDM-K64F MCU.

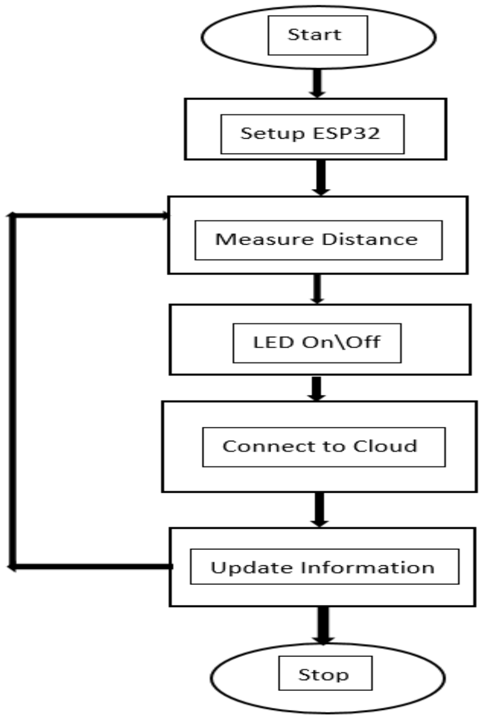


Figure 5.3.1: Software Design Flow Chart.

6.0 INTEGRATION AND TESTING

6.1 Integration Plan

The items tested consist of the integration of the software code modules developed as well as the hardware components. For testing software code modules, we choose the middle-out approach whereas for testing the hardware components of the subsystem, we choose the top-down approach. The integration sequence is described below, while the individual tests are in section 6.3.

6.1.1 Software Integration Sequence

The integration approach chosen for the software system was a middle-out approach. The reason for taking this approach was so that we could simultaneously begin testing the Model and View's functionality, even if one or the other was not already finished.

Though we did not do any explicit unit tests on our software, we did incremental tests of the components as we built the software to ensure all of the modules were working together as they should. This helped us remove defects early and focus on future integrations.

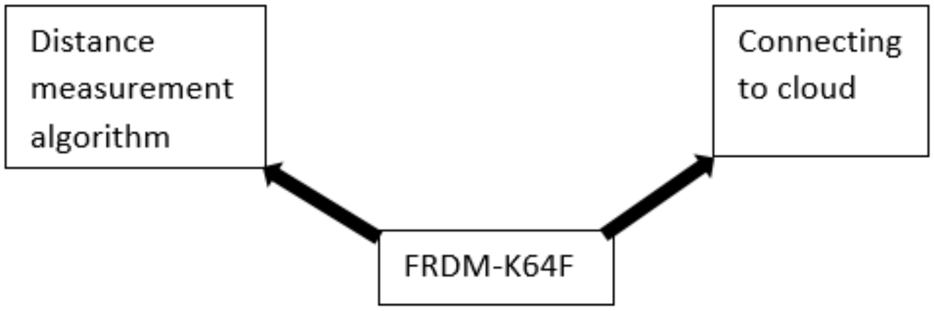


Figure 6.1.1.1: Software Integration Order for the ECIS System.

		Integration Sequence Order	Rationale
1.		Model	The 'Model' was the first component to be integrated because it models the data in which all other components use.
2.		Controller	The 'Controller' processes the model and is used by the view.
3.		View	The 'View' was integrated after the 'Controller' and the 'Model' because it relies on both of those components to function properly.
4.	a.	ESP8266	The ESP8266 was integrated before cloud server functionality because it is the bridge between the cloud server and ECIS hardware system.
	b.	Cloud Server	The 'cloud server' was the last component to be integrated because it is the final destination of the data.

Table 6.1.1.1: Software Integration Strategy for the ECIS System.

6.1.2 Hardware Integration Sequence

The integration approach chosen for this system was a top-down approach. This means that the designer has an overall vision of what the final system must do, and the problem is partitioned into components, or subsystems that work together to achieve the overall goal of the system for this project. Then each subsystem is successively refined and partitioned as necessary.

In the case of our ECIS system project, the overall objective was determined as aforementioned in the previous sections; the major subsystems are defined such as the NXP FRDM-K64F, HC-SR04 Ultrasonic Sensing Module, and ESP8266 Wi-Fi Module.

Before we integrated the various components of the subsystem, we performed individual testing of each hardware component. This way the project was built down starting from the top level because this method was more suitable, where it is unlikely that bringing together pieces in an ad-hoc fashion would have successfully solved the problem. This integration test, as described, is at the component level.

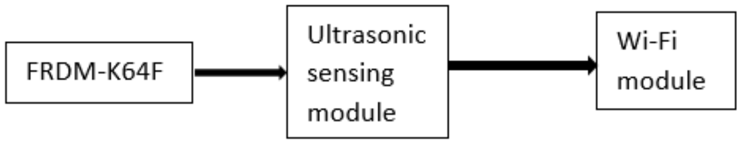


Figure 6.1.2.1: Hardware Integration Order for the ECIS System.

6.1.3 Full Hardware-Software Integration Sequence

The integration approach chosen for this system was a top-down approach. This means that the designer has an overall vision of what the final system must do, and the problem is partitioned into components, or subsystems that work together to achieve the overall goal of the system for this project. Then each subsystem is successively refined and partitioned as necessary.

Before we integrated various software components as aforementioned in section 6.1.1, we assembled all hardware components as per section 6.1.2 because integration of software build(s) depends on the hardware modules. In particular, all the hardware modules had to function properly before we could integrate software modules with hardware modules.

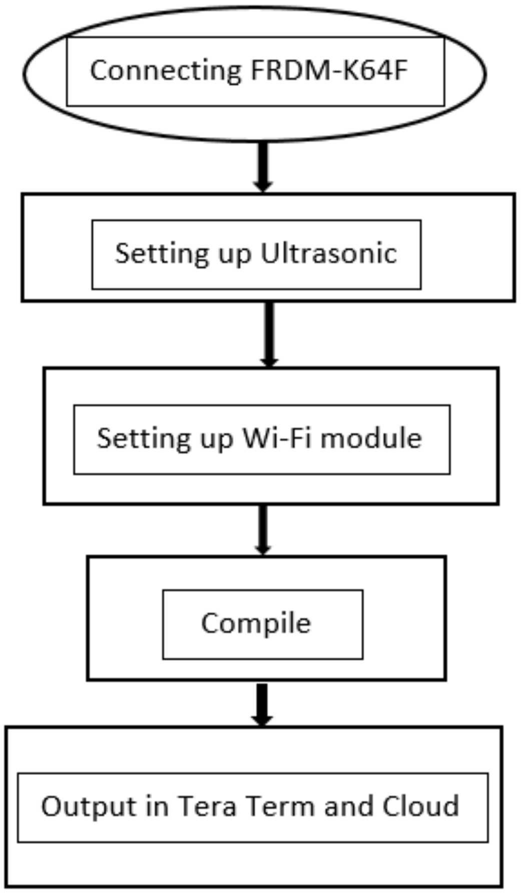


Figure 6.1.3.1: Full Hardware-Software Integration Order for the ECIS System.

Integration Sequence Order			Rationale
1.		NXP FRDM-K64F MCU	The 'NXP FRDM-K64F MCU' is the backbone of the entire system and provides support to various kinds of services, particularly including communication with the server, processing, and the terminal output, for the appropriate functioning of the entire system to meet the goals.
2.		Ultrasonic Sensing Module	Ultrasonic sensor is a key component of this project. It was integrated with the MCU before any other components. Echo, Trig, Vcc and GND pins were connected to the MCU's headers accordingly.
3.		Wi-Fi Module	ESP8266 Wi-Fi module was integrated after the sensor module because it facilitates communication with the MCU and the cloud server.
4.		Compiling SW	Next, our custom developed software was then compiled and flashed to the MCU.
5.	a.	Tera Term	Tera Term, an open-source terminal program, was then configured to read the output from MCU on to our computer's screen.
	b.	Cloud Server	Cloud server, ThingSpeak, was last to be integrated into our prototype build because it depends on all the other components to function as intended.

Table 6.1.3.1: Full Hardware-Software Integration Strategy for the ECIS System.

6.2 Final Functional Test

The final functional tests were completed after software was fully developed, all of the hardware had been assembled, and the two had been integrated together. The tests that were carried out are outlined as follows:

1. Action: **Place an object in front of the ultrasonic sensor.**
   Expected Result: Sensor should detect it and give an output on Tera Term and in cloud. Green LED should glow on the MCU.

2. Action: **Remove any objects, if placed, in front of the ultrasonic sensor.**
   Expected Result: An alert message should generate saying that the shelf is empty. Red LED should now glow on the MCU.
3. Action: **Place an object in front of the sensor in the daylight.**
   Expected Result: Sensor should detect it and give an output on Tera Term and in cloud. Green LED should glow on the MCU.
4. Action: **Place an object in front of the sensor in complete dark.**
   Expected Result: Sensor should detect it and give an output on Tera Term and in cloud. Green LED should glow on the MCU.
5. Action: **Turn of AP to which the system connects.**
   Expected Result: In Tera Term, a connection error should be displayed.

These tests are intended to confirm that the system has been successfully integrated, and now is ready for full system test by engineering. Section 6.3 provides the test results of the integration and testing carried out as aforementioned.

6.3 Testing Details with Results

6.3.1 Unit Tests

Test Writer: Priyank Kalgaonkar.			
Test Case Name:	Test `Sensor1.start()`.	**Test ID #:**	SW-UT-01
Description:	Verify that an object, which is placed 10 cm away from the sensor, is detected and output is shown in Tera Term, and Green LED is glowing. When the object is removed, Red LED should glow and alert message must be seen in Tera Term.		
Tester Information			
Name of Tester:	Sahil Kumar, Linknath Surya Balasubramanian	**Date:**	11/15/2019
Software Ver:	ECISsystem_ver.1.1	**Time:**	12:45 PM
Setup:	Setup the hardware, flash SW to MCU and then open and configure Tera Term.		

Step	**Action**	**Expected Result**	**Pass**	**Fail**	**N/A**	**Comments**
1	Object at 10cm.	Object detected correctly, Green LED glowing.	✓			Issue: No readings. Solution: Fixed errors in formula.
2	Object removed.	Alert shown in Tera Term. Red LED is glowing.	✓			Verified by inspection.
	Overall test result:		✓			`Sensor1.start()` has passed the test after troubleshooting errors.

Table 6.3.1.1 (Software-UT-01)

Test Writer: Priyank Kalgaonkar.						
Test Case Name:	Test `void Wifi_Tx(void)`.				**Test ID #:**	SW-UT-02
Description:	Verify that the ESP8266 Wi-Fi module is working as intended.					
Tester Information						
Name of Tester:	Linknath Surya Balasubramanian, Sahil Kumar				**Date:**	11/15/2019
Software Ver:	ECISsystem_ver.1.2				**Time:**	1:15 PM
Setup:	Setup the hardware, flash SW to MCU and then open and configure Tera Term and ThingSpeak cloud server portal.					

Step	**Action**	**Expected Result**	**Pass**	**Fail**	**N/A**	**Comments**
1	Input correct AP credentials.	ESP module should connect to AP and output should be shown in Tera Term. Cloud server portal should reflect the same.	✓			Issue: Server IP was incorrect. Solution: Changed IP address in code.
2	Input incorrect AP credentials.	ESP module should throw a connection error in Tera Term. Should not connect to internet.	✓			Verified by inspection.
	Overall test result:		✓			`void Wifi_Tx(void)` has passed the test after troubleshooting errors.

Table 6.3.1.2 (Software-UT-02)

6.3.2 Acceptance Test

Test Writer: Priyank Kalgaonkar.			
Test Case Name:	Full System Test	**Test ID #:**	FullTest-AT-01
Description:	Test the proper functioning of the final system as intended.		
Tester Information			
Name of Tester:	Sahil Kumar, Linknath Surya Balasubramanian, Priyank Kalgaonkar.	**Date:**	11/15/2019
System Ver:	ECISsystem_ver.1.4	**Time:**	02:05 PM
Setup:	This test shall be performed on the final system before validation. No special setup is required.		

Step	**Action**	**Expected Result**	**Pass**	**Fail**	**N/A**	**Comments**
1	Object at 10cm.	Object detected correctly, Green LED glowing. Results in Tera Term and cloud.	✓			Verified by inspection.
2	Object removed.	Alert shown in Tera Term. Red LED is glowing.	✓			Verified by inspection.
	Overall test result:		✓			No issues found.

Table 6.3.2.1 (FullTest-AT-01)

6.4 Performance Analysis

Our acceptance test has been carried out successfully. We encountered a few problems during the acceptance testing, but they were quickly fixed. One problem was due to the use of wrong IP address of our IoT cloud service provider. This was fixed by modifying the IP address in our code. We also discovered that our system would not output distance measured using the ultrasonic sensor. The issue was formula calculation which was quickly fixed.

7.0 ECONOMICS

The system development life cycle of our ECIS system project is composed of a number of clearly defined work phases, such as: brainstorming, design, implementation, testing and delivering the final product. Like anything that is prototyped, the system development life cycle aims to produce high quality systems to meet professional engineering standards, based on the need for an easy-to-use product within scheduled time frame and budget estimates.

7.1 Cost Estimation

The budget includes costs for all hardware requirements for building the prototype including the NXP FRDM-K64F MCU, sensors and other related components. Following is a table of cost of individual components:

Component	Price	Quantity	Total
NXP FRDM-K64F MCU	$59.82	1	$59.82
HC-SR04 Ultrasonic Sensor	$5.49	2	$10.98
ESP8266 Wi-Fi Module	$9.77	1	$9.77
Breadboard	$4.35	1	$4.35
TOTAL		5	**$84.92**

Table 7.1.1: Table of Cost Estimation

7.2 Engineering Labor Hours

In determining overall cost estimates, engineering labor hours also need to be taken into consideration. Table 7.2.1 gives an estimate on the labor hours for each activity below:

Activity	Labor (In Hours)
Meeting 4 days/week (3 group members)	48
Critical Design Review	5
Software Development	20
Hardware	5
System Integration and Testing	6
Finalize Implementations	20
Total	**104**

Table 7.2.1: Table of Engineering Labor (In Hours)

7.3 Specialized Facilities and Resources

Implementation of the ECIS system requires resources and a location where these resources are available. Table 9.4 shows the specialized facilities that were used for research and development of the project.

Facility	Location	Resources	Use
SL111: ECE533 Lab	Engineering Science & Technology (SL) Building at IUPUI.	Hardware such as voltmeters, oscilloscopes, soldering station, wires and breadboards.	Testing purposes.

Table 7.3.1: Table of Specialized Facilities and Resources

8.0 PROJECT MANAGEMENT PLAN

8.1 Organizational Responsibilities

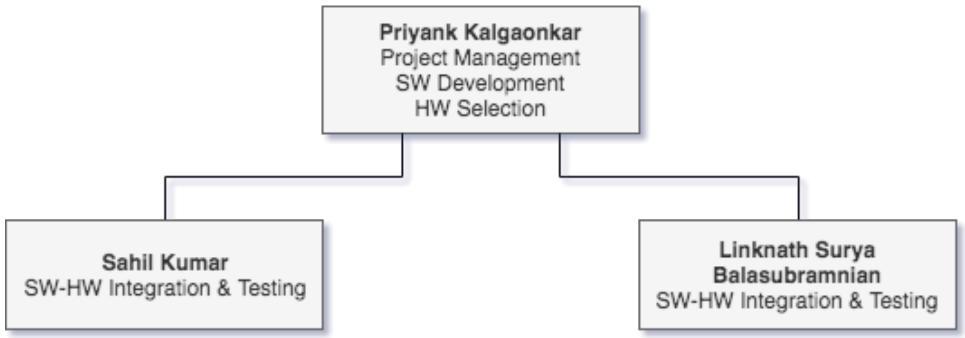

Priyank Kalgaonkar
Project Management
SW Development
HW Selection
Sahil Kumar
SW-HW Integration & Testing
Linknath Surya Balasubramnian
SW-HW Integration & Testing

Figure 8.1.1: Formal Organization Chart of Organizational Responsibilities

8.2 Work Breakdown Structure

ID	Activity	Description	Deliverables/ Checkpoints	Dura-tion (days)	People	Resources	Predec-essors
1	**Software**						
1.1	Acquire necessary software.	Ensure necessary software is installed successfully and ready to use.	• Identify necessary software. • Download/install software.	3	• Priyank • Surya	Personal Computer	
1.2	Brainstorm SW development ideas	Design software	• Detailed software flowchart.	1	• Priyank	Mbed SDK	1.1
1.3	Formulate Test Plan.	Determine plans for testing purposes.	• Unit test plans	1	• Priyank	Personal Computer	1.2
1.4	Develop Software.	Write code for ECIS system.	• MVC code • Setup cloud server	4	• Priyank	Mbed SDK	1.2 1.3
2	**Hardware**						
2.2	Complete Hardware Design.	Design hardware.	• Identify components. • System Schematic.	2	• Priyank • Sahil	• Internet.	1.1
2.3	Purchase K64F and Other Components.	HW procurement.	• Identify parts. • Place order. • Receive parts.	7	• Priyank		1.2

ID	Activity	Description	Deliverables/ Checkpoints	Dura- tion (days)	People	Resources	Predec- essors
2.4	Test individual components for hardware defects.	Test with supply voltage input.	• Test data showing appropriate functioning of indivi- dual components.	1	• Priyank • Sahil	• SL111. • Personal Computer.	1.2 1.3
2.5	Study and draw possible pin connections.	Study K64F and component pins.	• Test data showing appropriate output voltage.	1	• Sahil • Priyank	• SL111. • Oscilloscope • Datasheet.	1.1 1.3
3	**System Integration and Testing**						
3.1	Integrate all software and hardware components.	Assemble hardware and flash software.	• Test data showing desired overall functioning of the system.	3	• Surya • Sahil	• SL111. • Personal Computers.	1 2
3.3	Testing Assignment.	Create a test plan document.	• Unit test suite. • Integration strategy • Acceptance test suite.	1	• Priyank	• Book.	3.1
3.4	Improvements.	Identify defects and implement improvements.	• Identify defects. • Make necessary improvements.	1	• Priyank	• SL111. • Scanning system.	1 2 3
4	**Finalize Implementations**						
ID	**Activity**	**Description**	**Deliverables/ Checkpoints**	**Dura- tion (days)**	**People**	**Resources**	**Predec- essors**
4.1	Create Final Report.	Prepare report containing all necessary information about the project.	• Integration and testing results.	3	• Priyank	• Personal Computer.	1 2 3
4.2	Review and Submission.	Find and fix discrepancies in the document.	• Final report.	2	• Surya • Priyank		4.1
4.3	Final Presentation	• Final demo- nstration of the prototype to the general public.	• Working prototype.	1	• Priyank • Surya • Sahil		1 2 3

8.3 Risk Management

There were several risks identified with our project. The first major risk was that various project deadlines such as development, integration, testing, and final building of the prototype would not be met. This was mitigated by creating a detailed plan and setup efficient communication channels between the team members to manage our time correctly and efficiently.

Another risk was not having the critical components available in a timely fashion. Shipments could have easily been delayed, and parts could get damaged during shipping. This risk was avoided by organizing and selecting all the components necessary for our design as early as possible. Once this was complete, all the components were ordered at once. This gave enough time for complications and troubleshooting, which included parts that did not work. At first, the NXP FRDM-K64F that we received was faulty and the ultrasonic sensor was dead on arrival (DoA). Faulty components had to be replaced from the manufacturer with the same new components. Again, we avoided real complications by using good time management.

Additionally, our group ran the risk of not complying with specific standards for hardware components. We were able to avoid this risk by purchasing all hardware components off the shelf. This meant that our group also had to deviate from our original intentions of building our own housing unit to replicate a retail store shelf for the ECIS system and all of its components.

9.0 OTHER CONSIDERATIONS

9.1 Professional and Ethical Issues

Many of the Professional and Ethical issues regarding safety are out of scope for our project since all of the hardware components we are using are purchased off-shelf. However, there are several concerns that could be faced pertaining to the system failing. Some of the various ways the system could fail include: the store Wi-Fi malfunctions, any of our hardware parts breaking, or any piece of the system is hacked into. Our system will not be able to control the Wi-Fi malfunctioning since this is owned and maintained by the retail store (business). It also cannot control the hardware failure or misuse/abuse by the user.

Our group allowed for the systems already in place in stores to deal with all of the computer related security issues that could be encountered. For example, stores currently have their own systems for purchasing transactions which are up to security standards, so our group decided to keep purchasing out of our design. There is no critical information being stored on our system, such as credit card numbers or personal information. Therefore, we concluded that there would not be any malicious activity. As an extra precaution, SSL can be used to encrypt communications with the cloud server.

Additional security may be needed when communicating with the cloud service providing product pricing information. Though our current design does not take this into account, the addition of security features (such as SSL and authentication) is recommended for future work.

10.0 IMPACT OF ENGINEERING AND COMPUTING SOLUTION

To reduce the strain and increase efficiency, we are introducing the idea of inventory automation using electronically connected intelligent shelves. By allowing the on-site warehouse to monitor shelves in real-time and replenish depleting shelves when necessary, it will help the overall system to be more efficient which will ultimately help, both, the consumers and the retail business. This idea of ECIS system also opens door to predictive analysis using machine learning and data mining techniques, and for researching shopping trend of a particular product or product category.

Our design of ECIS system prototype includes an array of ultrasonic sensors, which can be retrofitted in existing shelves with minimal modifications or built-in in to new shelves, connected wirelessly to a central server from where the inventory of goods on the shelf can be monitored in real-time as well as data acquired from these sensors can be used to perform predictive analysis using data mining, feature engineering and machine learning techniques to better predict future product sales and minimize inaccurate forecasting instances as aforementioned.

Like other automated solutions, this project could result in some jobs being reduced or eliminated. Specifically, not as many store employees will be needed since only few employees will be needed to replenish shelves. However, employees would still be needed on the backend, in the warehouse, as well as to ensure the ECIS system is operating properly. These employees could also be reassigned to other jobs within the store that could help the store run more efficiently.

11.0 CONCLUSIONS AND RECOMMENDATIONS

11.1 Conclusions

The objective of the Electronically Connected Intelligent Shelves (ECIS) system is to automate the inventory replenishment process inside a traditional retail store by monitoring the stock of products on the shelf consistently throughout the day, provide real-time information, alert the staff when the stock on shelf is low as well as provide information for data analysis and research purposes.

The design of the ECIS system includes an array of sensors, each placed behind each row of products (consumer goods), centrally connected to a microprocessor unit, which then processes the data and sends it to cloud for alerting and data analysis purposes. Most modern shelves at a retail store are either made of metal or a combination of metal and plastic and are designed to nest into one another by drilling a two to three holes. This facilitates easy and quick installation of the sensors and system, and ultimately aids in shopping experience for consumers and increased revenue for retailers over the long run.

While there are alternate approaches to this problem, this solution has distinct advantages compared to these other approaches. One approach currently being pilot-tested by Amazon.com Inc., a multi-national e-commerce and technology company based in Seattle, Washington, incorporates a similar idea of electronic shelves using weight sensors as well as QR codes, video cameras and computer vision systems in its Amazon Go physical stores to increase efficiency throughput, reduce labor-costs, increase revenue and improve overall customer shopping experience.[3] This approach, although more accurate, is very expensive to implement and maintain for a traditional retail store due to multiple types of sensors, cameras and computer vision systems utilized in this system.

Another design of electronic shelves is the Smart Shelf by PointOfSale.com. In this system, weighing scales have been installed into the shelves. These weight sensors consistently notify the back-end system about the existing quantity of items on the shelves and sends an alert to the warehouse when the shelf is empty.[4] However, weighing scales need periodic maintenance and re-tuning, and cost of installation is high due to significant modifications required to retrofit traditional shelves.

In conclusion, the ECIS system satisfies the goal of providing a low-cost alternative for inventory automation using ultrasonic sensors. Customers will have a more enjoyable shopping experience as they can find the products they want to purchase, in-stock, and minimize the possibility of total cart abandonment and losing a customer's business as well as increase revenue by having more products on sale, for the retail businesses.

11.2 Recommendations

Although the ECIS system has its advantages, there is still room for improvement. The budget of the project as well as time hindered project development significantly. That being said, future research is always an option and there are several recommendations that can be made for this project to make its success greater.

One recommendation for future research and development is to use a combination of Infrared (IR) sensors since ultrasonic sensors work on the principle of reflection soundwaves that can be hindered factors such as humidity, temperature, but not very significantly. Hence, IR sensors will be ideal to use inside a cold storage shelf such as refrigerators.

Another suggestion is to connect databases of large-chain retail stores such as Walmart based on region to improve inventory management practices. For example: store A is low on product X and the store A's request to central distribution hub for more stock of product X is delayed / has not been satisfied yet. Instead of running out-of-stock on product X, store A can automatically start checking with other stores nearby and can request to temporary fulfill a certain amount of product X for the meantime. This can all be done through cloud's central monitoring portal by connecting all retail stores in the region under one umbrella.

12.0 REFERENCES

[1] Berger, R. (2002). Full-shelf satisfaction. *Reducing the out-of-stocks in the grocery channel. Grocery Manufacturers of America (GMA)*.

[2] Corsten D., Gruen T. (2005) On Shelf Availability: An Examination of the Extent, the Causes, and the Efforts to Address Retail Out-of-Stocks. *In: Doukidis G.J., Vrechopoulos A.P. (eds) Consumer Driven Electronic Transformation. Springer, Berlin, Heidelberg*, pp. 1-2.

[3] Boyle, Alan. “Fresh Patents Served up for the Smart Shelf Technologies Seen in Amazon Go Stores.” *GeekWire*, 5 Sept. 2018, www.geekwire.com/2018/fresh-patents-served-smart-shelf-technologies-seen-amazon-go-stores/.

[4] Team, Editorial. “Smart Shelf Technology Shapes Retailing.” :11 Mar. 2019, pointofsale.com/redefine-your-in-store-experience-with-smart-shelves/.

[5] Environmental Certifications. (n.d.). Retrieved from http://www.nxp.com/company/our-company/about-nxp/corporate-responsibility/environmental-compliance-organization/environmental-certifications:ENV_PBROAD.

[6] “Certification: Espressif Systems.” *Certification | Espressif Systems*, www.espressif.com/en/certificates.

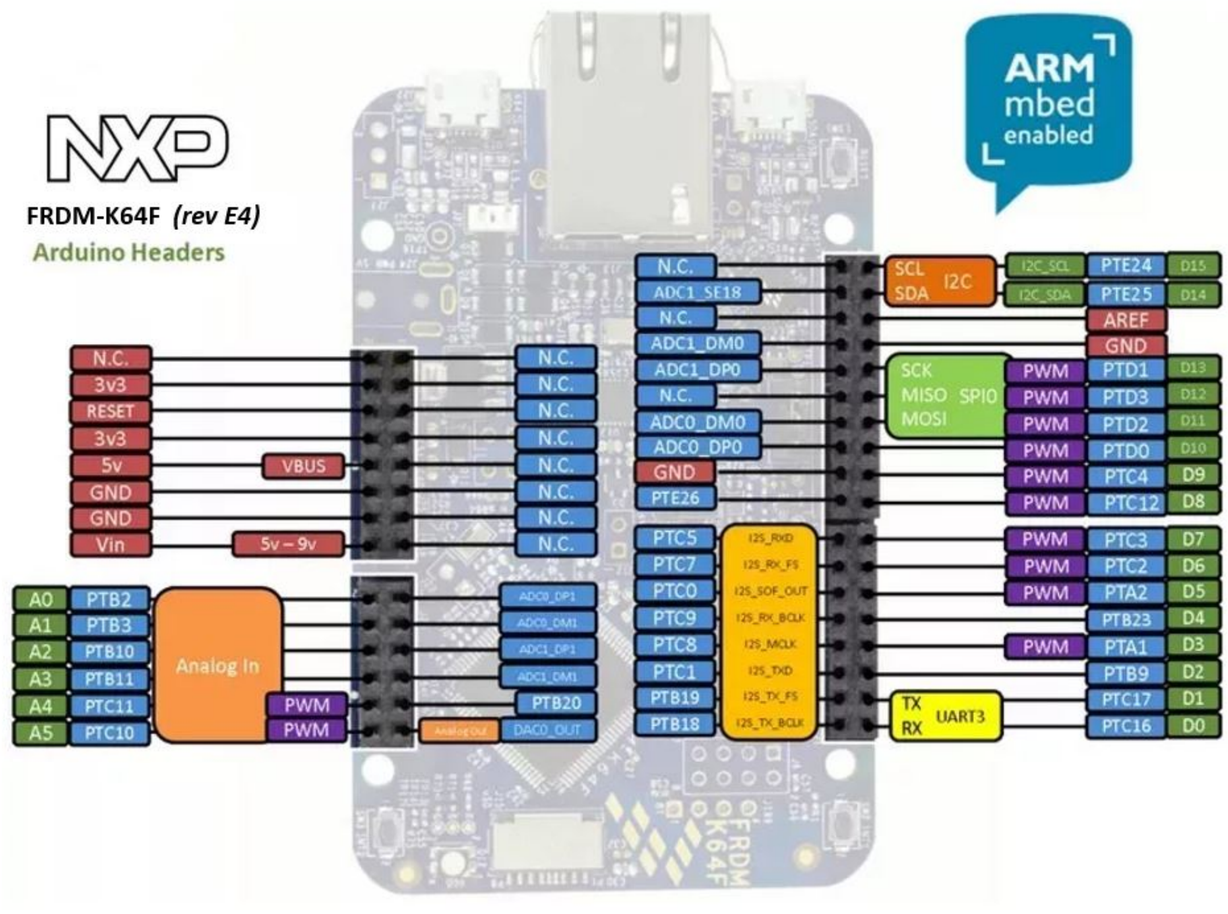
[7] Standards Compliance. (n.d.). Retrieved from http://www.sparkfun.com/compliance.

[8] mbed SDK - Handbook. (n.d.). Retrieved from https://os.mbed.com/handbook/mbed-SDK.

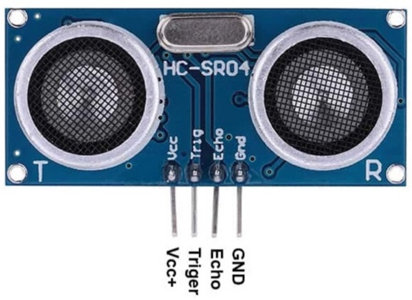
[9] Hearing range. (2019, December 5). Retrieved from https://en.wikipedia.org/wiki/Hearing_range.

Appendices

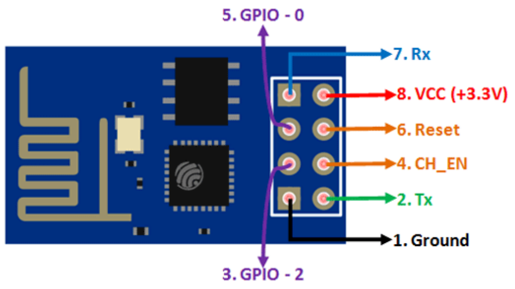
Appendix 1: Component Pinout Diagram



NXP FRDM K64F



HC-SR04 Ultrasound Sensor



ESP8266 Wi-Fi Module

Appendix 2: Program Source Code

```
/**
Group 1: Electronically Controlled Intelligent Shelves
Developed by: Priyank Kalgaonkar
ECE53301 - Final Project - Fall 2019
**/

#include "mbed.h"
#include "hcsr04.h"
#include "ESP8266.h"
#include "math.h"
#define CloudIP "184.106.153.149"   //IP Address of ThingSpeak Cloud Server

DigitalOut RLed(LED1);              //Onboard Red LED = Shelf Out of Stock
DigitalOut GLed(LED2);              //Onboard Green LED = All OK
DigitalOut BLed(LED3);              //Onboard Blue LED for Wifi Tx Indication
HCSR04 usensor1(D8,D9);             //ECHO Pin=D9, TRIG Pin=D8
Serial pc(USBTX,USBRX);
ESP8266 wifi(PTC17, PTC16, 115200); //Tx Pin:PTC17; Rx Pin:PTC17;
                                    //Baud rate:115200

void wifi_send(void);;

int distance1;
float dist_remaining, dist_percent;
char snd[255],rcv[1000];

int main()
{
    pc.baud(115200);
    pc.printf("Initial Setup\r\n");
    wifi.SetMode(1);
    wifi.RcvReply(rcv, 1000);             //Receive a response from ESP

    pc.printf("Conneting to WiFi\r\n");   //AP Setup Initialization
    wifi.Join("MCU", "IUPUIIUPUI");
    wifi.RcvReply(rcv, 1000);
    pc.printf("%s\n", rcv);
    wait(8);

    wifi.GetIP(rcv);                      //Obtains an IP address from the AP

    while (1)
    {
        wifi_send();

        BLed=1;
        wait(2.0f);

        BLed=0;
        wait(1.5f);
    }
}
```

```
void wifi_send(void)
{
    while(1)
    {
        pc.printf("Syncing Data with Cloud, Please Wait.\n\r");

      //Ultrasound Sensor (HC-SR04) Initialization
        int a = 30;
        usensor1.start();
        wait_ms(500);

      //Calculating Distance Percentage Remaining
        distance1 = usensor1.get_dist_cm();
        dist_remaining = a-distance1;
        dist_percent = (dist_remaining/30)*100;

        if (distance1<30) {
            RLed = 1;
            BLed = 1;
            GLed = 0;
            printf("\rPercent remaining: %f\r", dist_percent);
        } else {
            GLed = 1;
            BLed = 1;
            RLed = 0;
            printf("\rShelf Empty! Replenish Stock.\r");
        }

      //Sending Data to the Cloud Server via ESP8266 WiFi Module
      //AT+CIPMUX=0: Enable Single Channel Mode
        strcpy(snd,"AT+CIPMUX=0\n\r");
        wifi.SendCMD(snd);
        wait(1);
        wifi.RcvReply(rcv, 1000);
        wait(1);

      //Establish TCP connection w/ Cloud Server
        sprintf(snd,"AT+CIPSTART=4,\"TCP\",\"%s\",80\n",CloudIP);
        wait(1);
        wifi.RcvReply(rcv, 1000);
        wait(1);

      //Set length of the data that will be sent
        strcpy(snd,"AT+CIPSEND=100\n\r");
        wifi.SendCMD(snd);
        pc.printf("%s\r", rcv);
        wait(1);
        wifi.RcvReply(rcv, 1000);
        pc.printf("%s\r", rcv);
        wait(1);

      //Pushing the data acquired from HC-SR04 Ultrasonic Sensor to Cloud
        sprintf(snd,"GET
                https://api.thingspeak.com/update?api_key=3LV9GHWHJYDZNWEH&fie
                ld1=%f \r\n\r\n", dist_percent);
        pc.printf("%s\r",snd);
        wifi.SendCMD(snd);
        wait(1);
        wifi.RcvReply(rcv, 1000);
        pc.printf("%s\r", rcv);
    }
}
```