Laundry Now

Frank Monforte, Ryan Russell

Electrical Engineering Senior Design Project

http://www.laundrynowssu.weebly.com

Advisor: Dr. Farid Farahmand, Client: Tramaine Austin-Dillon RLC

Submitted to the Department of Engineering Science in partial fulfillment of the requirements for the Bachelor of Science degree in Electrical Engineering at Sonoma State University – May 2015

1 ABSTRACT

“Laundry Now” is a system that is intended to make the process of doing laundry more efficient. It is designed to be used in apartment complexes or dormitory housing laundry rooms and it will allow residents to easily check if washers or dryers are available or in use. This system will also report the number of cycles each machine has completed, allowing property management to see which machines are being used and how much. By keeping track of the number of cycles a machine has completed, Laundry Now can help facility management determine when repairs or maintenance may be needed, and make a maintenance schedule to keep machines operating at peak efficiency while anticipating potential problems before they happen.

2 ACKNOWLEDGEMENTS

We would like to thank the following people for their assistance in the planning, design, and implementation of the Laundry Now project.

Dr. Farid Farahmand	Advisor/Associate Professor
Tramaine Austin-Dillon	Industry Mentor
Dr. Haider Khaleel	Assistant Professor
Shahram Marivani	Adjunct Professor
Jeff Booher-Kaeding	Student Assistant

3 TABLE OF CONTENTS

1 Abstract 1
2 Acknowledgements 1
4 List of Figures 5
5 List of Tables 8
6 List Of Abbreviations 9
7 Introduction 10
8 Problem Statement 10
9 Problem Solution 10
10 Literature Review/ Existing Patents 10
11 Marketing Requirements 12
12 Design Approach 13
12.1 Project Overview 13
12.2 Demonstration Unit 13
12.3 System Operations 15
12.3.1 Server Components 15
12.3.2 Server COM Port – Diagnostic Information 15
12.3.3 HTTP Server Network Information 16
12.3.4 HTTP Server 17
12.3.5 Port Forwarding 18
12.3.6 Webpage Information 18
12.3.7 Webpage Layout 19
12.3.8 Machine Identification 21
12.3.9 QR Code 21
12.3.10 HTTP Server Operation 22
12.3.11 Real Time Clock 22
12.3.12 Day of the year 23
12.3.13 SD Card 23
12.3.14 Data Format Cycle Information 23
12.3.15 Data Import 25
12.3.16 Server – Controller Communication 26
12.3.17 Connectors 27
12.3.18 Server UART 28

12.3.19 Setup Controllers 29
12.3.20 Update Data 29
12.3.21 Server – Controller Data Bits 30
12.3.22 Controller Node Components 31
12.3.23 Sensor Node Components 33
12.3.24 Series SPI Communication 34
12.3.25 Overview of Controller Algorithm 35
12.3.26 Determination of the Number of Sensors 36
12.3.27 Three Axis Averaging 36
12.3.28 Error Checking 36
12.3.29 Dryer Algorithm 36
12.3.30 Washing Machine Algorithm 37
13 Potential Problems 38
13.1 False Trigger 38
13.2 Machines Too Quiet 38
13.3 Wi-Fi Router Not In Range 38
13.4 Users Not in Range of Wi-Fi 38
13.5 Unable to Access Back of Machine 38
13.6 Loose Connection Due to Continuous Wear 39
14 Failure Analysis & Changes 39
14.1 Loss of SPI data communication on PCBs 39
14.2 SD Card 40
14.3 WPS (Wi-Fi Protected Setup) 40
15 Economic Analysis 41
16 Social and Environmental Impact 42
17 Timeline and Gantt chart 42
18 Future Work 45
19 Summary 45
20 Reference 46
21 Appendices 48
21.1 Detailed Budget 48
21.2 Test and Measurements 49
21.2.1 Initial Proof of Concept 49

21.2.2 Bench Testing 50
21.2.3 On site Testing 71
21.3 Flow Charts 80
21.4 Circuit Diagrams 88
21.5 Bill of Materials 93
21.6 In-depth Description of Technology and Components 94
21.6.1 ADXL345 94
21.6.2 PIC18F45K20 94
21.6.3 PIC32MX69F512L on a Diligent Breakout Board 95
21.6.4 74HC7541 95
21.6.5 74HC138 96
21.7 Images of Final Project 97

4 LIST OF FIGURES

Figure 1: System Block Diagram ……..13
Figure 2: Demonstration Unit ……14
Figure 3: Webpage page……19
Figure 4: QR Code ……..21
Figure 5: LAUN_NOW.txt File formatting ……25
Figure 6: Microcontroller Diagram ……27
Figure 7: Server Wiring Layout ……..28
Figure 8: Mapping Diagram of Bits ……...31
Figure 9: Controller Wiring Diagram ……32
Figure 10: Sensor Wiring Diagram ……34
Figure 11: Controller Node to Sensor Node Connection ……..35
Figure 12: Generating of Test Information ……..49
Figure 13: Testing Graphs ……50
Figure 14: Device ID Request and Transmit ……...52
Figure 15: Status Register Request ……53
Figure 16: Status Register Transmission ……54
Figure 17: Overview Setup Communication ……55
Figure 18: Initial Register Location in Setup ……56
Figure 19: Data being placed in Initial Register ……..57
Figure 20: Benchmark, One Sensor, 0 ft Distance ……58
Figure 21: Test, 1 Sensor, 10 ft Distance ……59
Figure 22: Test, One Sensor, 10 ft Distance, Error Occurring ……60
Figure 23: Test, One Sensor, 10 ft Distance, Error Effect on Communication ……..61
Figure 24: Test, Eight Sensors, Buffers, 0 ft Distance ……..62
Figure 25: Test, Eight Sensors, 0 ft Distance, Delay Added Initial Reading ……63
Figure 26: Test, Eight Sensors, 0 ft Distance, Delay Added Second Reading ……64
Figure 27: Test, Eight Sensors, 10 ft between each Node, Total 80 ft Distance ……..65
Figure 28: Benchmark, Four Controllers, 0 ft Distant ……66
Figure 29: Benchmark, Four Controllers, 0 ft Distant, Overlapped ……67
Figure 30: Test 1, Four Controllers, 10 ft Distant ……...68

Figure 31: Test 1, Four Controllers, 10 ft Distant, Overlapped ……68
Figure 32: Test 2, Four Controllers, 10 ft Distant ……..69
Figure 33: Test 2, Four Controllers, 10 ft Distant, Overlapped ……69
Figure 34: Test 3, Four Controllers, 10 ft Distant ……...70
Figure 35: Sensor Mounting Locations ……74
Figure 36: Full Wash Cycle ……75
Figure 37: Zoomed-In Look at the Wash Cycle ……..75
Figure 38: Dryer Averages 0, 10, 20 ……76
Figure 39: Dryer Averages 50, 100, 200 ……77
Figure 40: Washing Machine Averages 0, 10, 20 ……..77
Figure 41: Washing Machine Averages 50, 100, 200 ……78
Figure 42: Test Results for Stacked Machines ……...79
Flow Chart 1: Higher Level Server System Block Diagram ……..80
Flow Chart 2: UART Initial Request ……..81
Flow Chart 3: UART Data Request ……..82
Flow Chart 4: Higher Level Controller Node Block Diagram ……83
Flow Chart 5: Three State Diagram ……..83
Flow Chart 6: Three State Dryer Algorithm ……84
Flow Chart 7: Three State Washer Algorithm ……85
Flow Chart 8: Webpage Build Process ……86
Flow Chart 9: Three Axis Reading ……87
Figure 43: Server Schematic ……88
Figure 44: Controller Node Schematic ……89
Figure 45: Sensor Node Schematic ……..90
Figure 46: Server PCB Layout ……..91
Figure 47: Controller Node PCB Layout ……..91
Figure 48: Sensor Node PCB Layout ……...92
Figure 49: Server PCB, Top View ……97
Figure 50: Server PCB, Bottom View ……...97
Figure 51: Controller Node PCB, Top View ……98
Figure 52: Controller Node PCB, Bottom View ……...98

Figure 53: Sensor Node PCB, Top View ……99
Figure 54: Sensor Node PCB, Bottom View ……99
Figure 55: Straight Ethernet Cable ……100
Figure 56: Test Board ……...101
Figure 57: Final Product ……...102
Figure 57: Final Poster and Project ……103

5 LIST OF TABLES

Similar Product Comparisons ……12
Marketing Requirements……12
Initial Time Line ……42
Initial Gantt Chart ……43
Final Time Line ……43
Final Gantt Chart ……44
Project Milestone ……44
Laundry Now Budget ……..48
Laundry Now Device Supplies ……..48
Laundry Now Bench Testing ……50
Laundry Now On Site Test Results ……..71
Bill of Materials ……93

6 List Of Abbreviations

SPI: Serial Peripheral Interface (Communication Interface)

MCU: Microcontroller Unit

UART: Universal Asynchronous Receiver/Transmitter

I2C: Inter-Integrated Circuit (Communication Interface)

HTML: Hyper Text Markup Language

HTTP: Hyper Text Transfer Protocol

MPIDE: Multi-Platform Integrated Development Environment

RAM: Random Access Memory

BTN1/2/3: Button 1/2/3 on the WF32

PC: Personal Computer

USB: Universal Serial Bus

SD: Secure Digital (Nonvolatile Memory Card)

COM: Communication (Port)

SSID: Service Set Identifier (Wi-Fi Network Name)

WPA2: Wi-Fi Protected Access Version 2 (Network Encryption)

IP: Internet Protocol

MAC: Media Access Control

URL: Uniform Resource Identifier

ISP: Internet Service Provider

ROM: Read-Only Memory

QR: Quick Response (Code)

RTC: Real Time Clock

RJ45: Registered Jack (45) (Connector Type)

TX: Transmitter

RX: Receiver

MSSP: Master Synchronous Serial Port

MEMS: Microelectromechanical System

LAN: Local Area Network

7 INTRODUCTION

The purpose of “Laundry Now” is to create a system that helps residents efficiently use their time when doing their laundry. This system will allow users to predetermine if there are washing machines and dryers that are available for use before having to leave their home or their room in a dormitory. This is to help users determine the best time to do their laundry. Laundry Now also tracks machine usage data allowing apartment management and maintenance teams to more accurately monitor machines in order to keep them operating at peak efficiency.

8 PROBLEM STATEMENT

In many laundry rooms in college dorms and apartment complexes, the only way to tell if a washer or dryer is in use is by going to the laundry room and physically checking if there is a machine available. It can be frustrating and time consuming having to constantly check the laundry room when waiting for a machine to become available.

9 PROBLEM SOLUTION

Laundry Now will allow users to check whether there is a washer or dryer available before leaving their home. The system will monitor accelerometers that will be attached to the washers and dryers, and when motion is detected the system will update itself to show that the machine is in use. When a user checks the website from their computer or mobile device, they will see the most recently updated statuses of the machines in the laundry room. Laundry Now will be non-invasive and simple to install because it only requires a single sensor node to be attached to each unit. Due to the non-invasive design, Laundry Now will be inexpensive to purchase and implement, and can be used with both newer and older washers and dryers.

10 LITERATURE REVIEW/ EXISTING PATENTS

Currently there are two different products commercially available that are similar to Laundry Now. The first is LaundryView, created by Mac Gray, and the second is

Remote Laundry Monitoring System, created by LG. Both of these systems are similar to the product that we are designing, but they each have significant limitations for our targeted market [1,3].

LaundryView is a system that was developed by Mac Gray as another option that works with the washer and dryers that they supply to colleges and apartment complexes. LaundryView allows users to predetermine if there are any washers or dryers that are available for use. They can do this by checking LaundryView's website that is continuously updated. This system also allows users to set a notification system that will send a message when the machine has finished,or is about to finish, the wash cycle. The major limitation of this system is that it requires the washers or dryers that Mac Gray supplies, creating the need to purchase entirely new machines in order to use the system. This significantly increases the financial burden of implementing this system [1,2].

The other system that is similar to Laundry Now is LG's Remote Laundry Monitoring Service, which is a device that allows homeowners to view the amount of time left on the current wash cycle. This device requires the installation of modems into a washer and dryer which will connect with the timer, and then will relay the information over the household power lines to the receiver, which needs to be plugged into the wall. This system works well for a single household, but it is not scalable for use in a facility with many machines [3].

Similar Product Comparisons

Device (1,2,3,4)	Need extra Equipment	Scalable	Internet Capable	Estimated Cost
LaundryView	Mac Gray supplied machines	Yes	Yes	Depends on the number of machines needed
LG's Monitoring System	No	No	No	$90.00
Laundry Now	No	Yes	Yes	$525 (Full System Parts); $160 (Demo Unit Parts)

11 MARKETING REQUIREMENTS

Function	Allows users to predetermine if there are any washers/dryers that are available for use, by checking the website.
Number of Machines	8 sensors per controller, 4 controllers per server, up to 32 different machines
Wi-Fi Connection	WPS connection
Powered	wall socket
Consumer Cost	$525 (Full System Parts); $160 (Demo Unit Parts)
Size	Server 4" by 4", controller 4" by 4", sensor 2" by 2"
Package	Plastic shells
PC and Cell Phone capable	Any device that is internet enabled
Sensor connection	Non-invasive, attached with double stick tape users might be reluctant as this will damage the finish on the machine if left for a long time.
Update Time	The webpage will be updated every 5 minutes to reflect the latest washer/dryer status.

12 Design Approach

12.1 Project Overview

This product will allow users to see the number of washers and dryers that are in use or available at their laundry facility without having to leave their home. It will do so through sensor nodes that are placed on every washer and dryer and are connected to the Serial Peripheral Interface (SPI) of the controller node microcontroller. The controller node will monitor the movements of the sensors to keep track of which machines are in use. The status of the machines will be stored on the individual controller nodes and the sensors will be checked every 2 minutes. The server will request the latest machine status and update the website every 5 minutes. When a request is received from a user over Wi-Fi, the server will return the HTML website that displays the latest machine status. See Figure 1 for the basic system design.

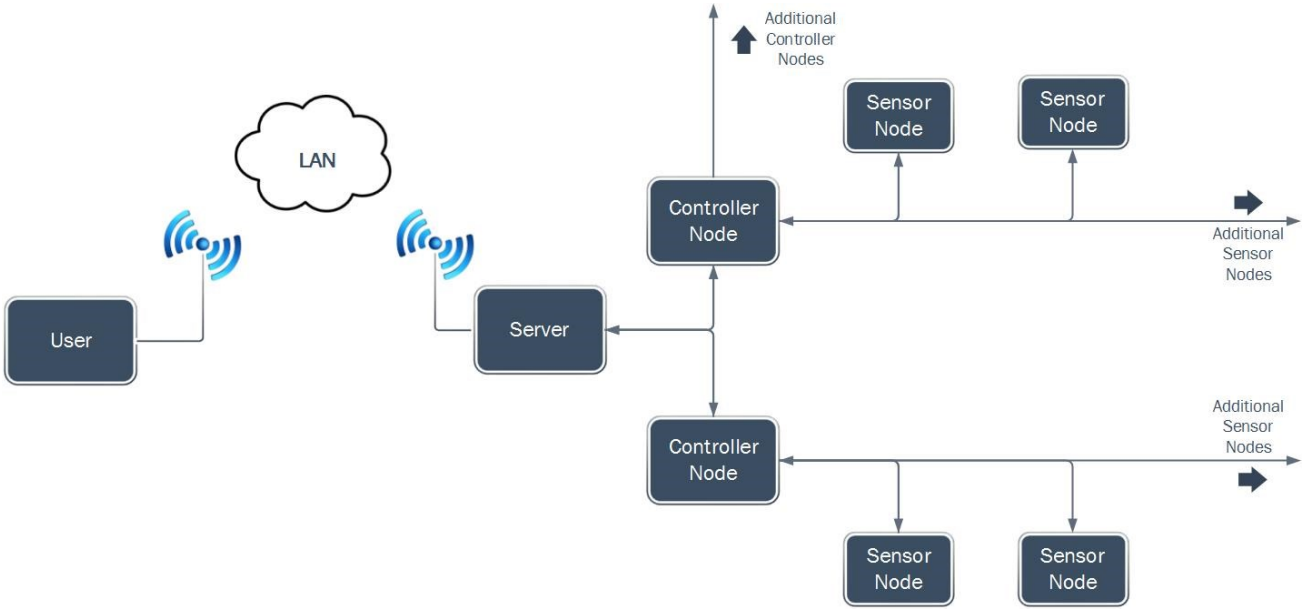


Figure 1: System Block Diagram

12.2 Demonstration Unit

As proof of concept, we are building a demonstration unit that utilizes the different parts of this design. The demonstration unit will have one server microcontroller controlling the system. There will be two control nodes (the system will be capable of supporting a

maximum of four controller units) with two sensor nodes connected to each controller node unit (the system will be capable of supporting a maximum of eight sensors per controller). The demo unit will require a single available wall outlet power source to power the server and connected components. There will also need to be an available Wi-Fi network provided by a router that supports WPA2 Wi-Fi encryption keys.

Laundry Now will be able to detect whether the washer or dryer is in operation or not. Because of its non-invasive design, it will not be able to directly detect if the load has actually been removed from the machine. By analyzing the movement data it will make the best determination of whether the machine is in use (Running), has completed a cycle (Just Finished), or is empty (Available). Just Finished will display on the webpage for five minutes then the machine's system state will change to Available. This information will then be sent to the server which will update the status of the machines accordingly. See Figure 2 for the system block diagram of the demonstration unit.

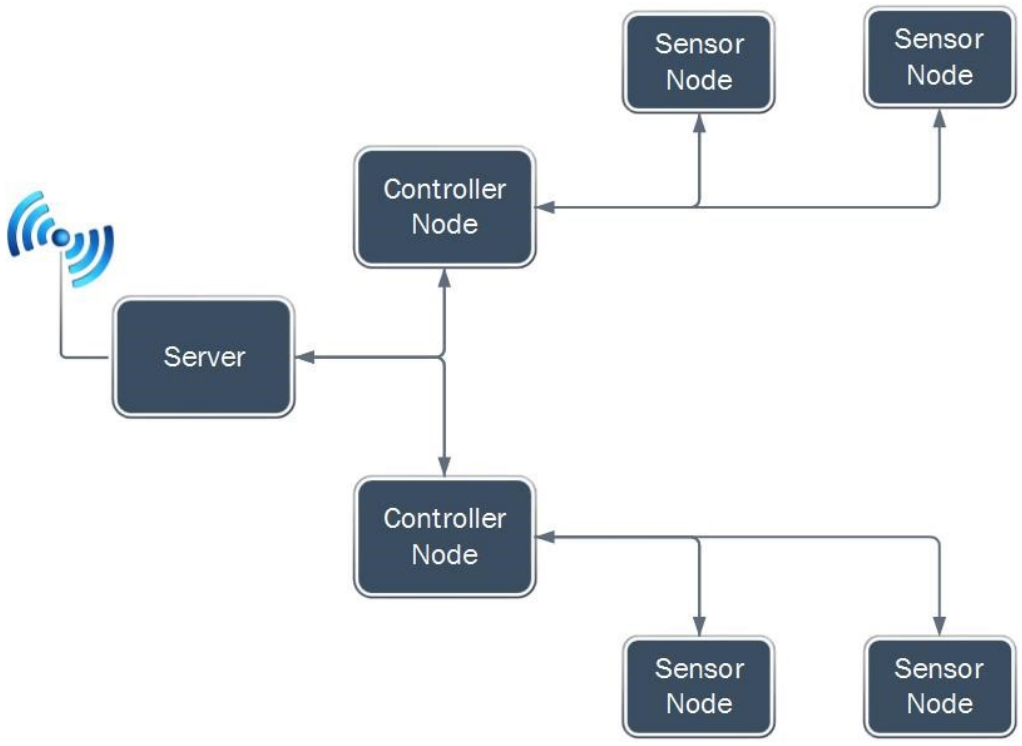


Figure 2: Demonstration Unit

12.3 SYSTEM OPERATIONS

12.3.1 Server Components

The server is a Digilent chipKit WF32 microcontroller board that features a Microchip PIC32MX695F512L MCU and a Microchip MRF24WG0MA Wi-Fi module. This board was chosen based on the combination of the Microchip PIC MCU and the Microchip MRF Wi-Fi module being built in to one board. The WF32 uses Digilent's MPIDE programming environment allowing for the use of chipKit, Arduino, and PIC XC32 commands and libraries. The HTTP server portion of the system is based on Digilent's supplied HTTP server examples and libraries that have been modified to fit the specific needs of the Laundry Now system. The PIC32 MCU is running at 80MHz (the default setting of the chip) and is continuously monitoring port 80 on its assigned IP as well as periodically gathering updated status information from the controller nodes (for more detailed information on data gathering see server controller UART section).

There are three buttons on the WF32 board, BTN2 and BTN3 are unused, and BTN1 is a master clear software reset that will clear the RAM and start the server from the top of the program. BTN1 will not cycle the power, the controllers and sensors will remain running as normal and will not see any change in operation when BTN1 is pressed. To cycle the power for the entire system, the power cord must be removed for 10 seconds, then plugged back in. This will reset the server, controllers, and sensors at once and force the entire system to reboot.

The WF32 board includes an SD card that is used to backup machine history information. The board has two UART modules used to communicate with a PC (optional, over microUSB) and the controller nodes (data gathering). There is an I2C module that is used to connect with the real time clock chip that is used as part of the data backup system. The WF32 board features other built-in components that are not implemented as part of this system.

12.3.2 Server COM Port – Diagnostic Information

The WF32 board has three USB ports on board: Standard-A Receptacle, micro-B plug, and mini-B plug. The Standard-A receptacle and micro-B plug USB ports are not used by the Laundry Now system. The mini-B plug is connected to the UART module on the

PIC32 and when connected to a PC running a serial port monitor, the Laundry Now server will output diagnostic text information about the status of the system.

When a USB cable is connected and a serial monitor program is started, the Laundry Now server will reboot. The power is NOT fully cycled in this case (software master clear reset), so the controllers will not reset and will continue to monitor machines as usual. As the server reboots it will reload the backed up data from the SD card, will attempt to reconnect to the Wi-Fi network, and will begin normal operation (See HTTP Server Operation section for more information).

There are a number of different diagnostic messages that can be printed to the screen through the COM port. This can be important for diagnosing Wi-Fi network problems, server-controller node communication errors and controller node- sensor node communication errors.

12.3.3 HTTP Server Network Information

Currently the Laundry Now server is only setup to support WPA2 Passphrase security keys. The implementation of additional network security types is included in the future works section.

The SSID and WPA2 key is stored in a plain text file saved on the SD card. By default this file is hidden (Windows OS) in a file titled “LAUN_NET.txt”. This information is imported from the SD card upon initial startup. If no SD card is present, or the SD card for some reason cannot be read, the network information will not be imported and a connection to the Wi-Fi network will not be established. The sever will timeout from searching for a valid network if valid data is not imported. The UART COM port (via microUSB) will print the network import status information.

If incorrect network information is entered into the file, the server will fail to connect with the network. It will periodically search for a network matching the supplied network information. The exact time between searches varies depending on how many failed attempts have occurred. Each time a failed attempt occurs the time between network searches increases.

If the Wi-Fi network is down, or goes down while the server is connected to it, the server will attempt to find and reconnect to the network. Each time a failed attempt to connect occurs, the time between network searches increases.

There is a supplied program loaded onto the SD card that will allow the user to easily setup the file containing the network information. Windows is currently the only supported OS for the network setup program. The program is titled “LNOW” on the SD card. The SD card must be plugged into a Windows PC and run directly from the SD card in order for the generated file to be saved in the correct location. The program asks the user first for the SSID, then for the WPA2 passphrase. Once this information is gathered from the user, the program searches for an already existing network information file (“LAUN_NET.txt”) and deletes it if present. It then creates a new file with the provided network information.

The file can manually be setup without the program by entering in the information through a standard text editor. The network information file “LAUN_NET.txt” is hidden by default (Windows OS). The first line of the file corresponds with the SSID, terminated by a carriage return and new line character (the “Enter” key will accomplish this though it is important to note the “\r\n” termination must be present for the program to properly import the information. The second line is the WPA2 passphrase, also terminated by a carriage return and new line character. The “\r\n” termination must be present at the end of both the SSID and WPA2 passphrase line in order for the information to be properly imported. Any data in the file after the WPA2 passphrase “\r\n” termination characters will be ignored.

12.3.4 HTTP Server

When the server attempts to connect with the Wi-Fi network, it requests the local IP address of 192.168.0.195 by default. Generally this IP is granted by the router as long as there are no other devices on the network that have already been assigned that IP. At this time, the IP address that the server requests is not configurable. A configurable IP address option is part of future works and would be implemented as part of the network information file that is stored on the SD card. It is recommended that in the settings of the router (most routers have this option) the MAC address of the server

board be setup to have a reserved IP address of 192.168.0.195. This will insure that another device is not assigned that IP, and the server will always be granted that specific IP address upon establishing a connection.

The chipKit WF32 board is configured to allow for up to six clients to be connected to the server at once. This is not a user configurable setting and cannot be changed.

To access the server on the local network, users must enter the IP address (192.168.0.195) into their browser URL window. This will send a request to the server on port 80 (HTTP) and the server will respond with the webpage. The server only has one page, and all of the information is displayed at once. If access to the server is needed outside of the local network the server is connected to, port forwarding must be setup. See the Port Forwarding section for more information.

12.3.5 Port Forwarding

In order for the website to be accessed, the user must be on the same Wi-Fi network as the sever unless port forwarding is setup. Port forwarding is a feature built into some routers – not all routers will have this option. In the settings of the router, port 80 must be setup to forward to the IP and/or MAC address of the server. Users can then enter the public IP address of the network the server is connected to (assigned by the ISP) and the router will forward port 80 (HTTP) requests to the Laundry Now server, which will then return the webpage to the router and the router will forward the webpage back to the user that requested it.

12.3.6 Webpage Information

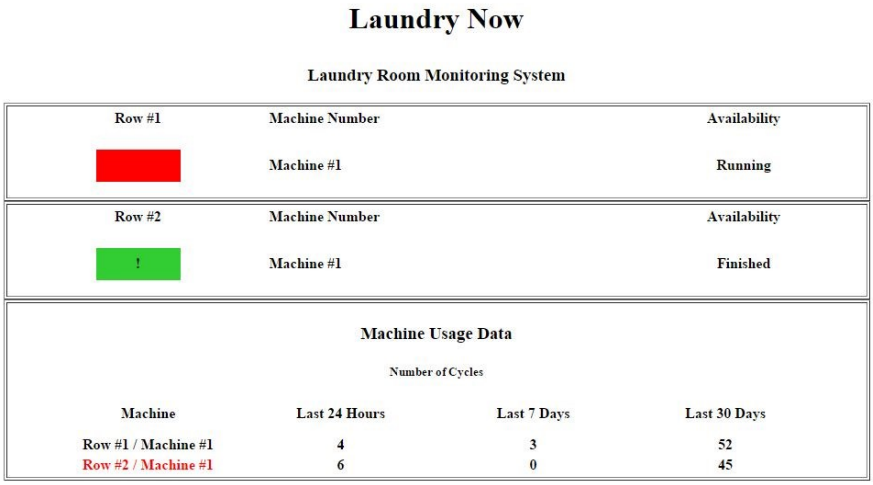
A template of the webpage is hardcoded into the ROM of the PIC32 and is built dynamically each time a client requests the page. The website is coded in plain HTML text using tables, and basic color and formatting settings.

When the server starts up it automatically determines how many controller nodes are connected, and how many sensors are connected to each controller (See controller setup operation for more information). This is used to determine the number of controller and sensor tables to build into the final page.

When a client requests the page, the server places the HTML page header into a temp file. It then places the header for the first controller. The server then looks at how many sensors are connected to the controller, and places that number of sensor headers into the temp file. The server looks up the latest status information for that particular machine and the correct colors and titles are filled in. Once the status information is placed into the temp file it closes the table for the first controller. This operation repeats for the rest of the controllers connected to the server until all controller and sensor templates are placed in the temp file. The tables for the machine cycle information are the same as above. The server then ends the HTML file with the closing tags for the tables and formatting and the page is returned to the user. See flowchart 10 for a visual representation of this process.

12.3.7 Webpage Layout

An image of the basic webpage layout for 2 controllers, each with 1 sensor, connected to the Laundry Now server. The various colors and codes are shown below along with an explanation. Figure 3 shows the webpage layout.



Laundry Now

Laundry Room Monitoring System

Row #1	Machine Number	Availability
	Machine #1	Running

Row #2	Machine Number	Availability
!	Machine #1	Finished

Machine Usage Data

Number of Cycles

Machine	Last 24 Hours	Last 7 Days	Last 30 Days
Row #1 / Machine #1	4	3	52
Row #2 / Machine #1	6	0	45

Figure 3: Webpage

The top half of the webpage shows the last known information for the machines connected to the Laundry Now system. Each sensor is organized into tables underneath the controller it is connected to.

The first column shows a box with a solid color. This corresponds with the current status of the machine listed under the availability column. The next column is the “Machine Number” column and lists the machine number so that it can be identified by users when they are in the laundry room. The third column lists the availability of the machine. The wording used here corresponds with the color shown in the first column and the codes are listed below.

Green / “Available”: The controller node has determined the machine is not running, and is available for use.

Red / “Running”: The controller has determined the machine is running, and is not available for a new user.

Yellow / “Just Finished”: This status is determined by the server, and is shown for 5 minutes after a machine has changed from “Running” to “Available”. During this time, the controller has determined the machine is no longer running, but the server displays this status to let users know that the machine may still be occupied.

Any Color with “!”: If there is an exclamation mark shown within the color box (of any of the above colors) there may be a problem with the machine. This information is for use by the management team or maintenance. It is shown when there has been no activity on a particular machine for the last 7 days. When this error message occurs, the row/machine number under the cycle data section will also be highlighted in red.

Blue: “-SEN/CON ERROR-“: If this status is shown, there is a communication error with either a controller or sensor. If this error is shown for only one machine on a controller/row, then it is most likely an error with a single sensor. If this error is shown on every machine on an entire row, it is most likely a communication error with a controller.

The bottom half of the website displays cycle information. The latest cycle information is updated by the controller every five minutes along with the machine status information. The week and month counts are only updated once every 24 hours (at midnight). This data save is triggered by the real time clock. The “Last 24 Hours” count is updated every five minutes with the status information. A single “count” is incremented when a machine status has changed from “Running” to “Available.”

12.3.8 Machine Identification

On the website, the machines are organized first by the controller they are connected to, then by sensor number. Sensor number is determined by the address that is selected on the sensor node itself. The controller number (or “Row” as shown on the website) is determined by the order in which the controllers are connected to the server. Controller Node number 1 corresponds with “Row A” on the website, controller node number 2 corresponds with “Row B” on the website, controller node number 3 corresponds with “Row C” on the website, and controller node number 4 corresponds with “Row D” on the website. This ordering is done so that in the laundry room each row can be labeled with a letter, and each machine with a number in order for users to identify which machine on the website corresponds with which machine in the laundry room.

12.3.9 QR Code

As a second option to access the website, a QR code can be generated to make it easier to get to the Laundry Now webpage. The QR code would link to either the local IP address of the server (192.168.0.195) or if port forwarding is setup, to the public IP of the network the server is connected to (assigned by the ISP). The QR code would allow a user to scan the barcode with their smartphone and it would automatically direct them to the IP address of the Laundry Now server, removing the need to enter in the exact numbers into the URL bar. This barcode could be placed in the laundry room on a bulletin board or another common place where people could easily scan, making the system easier to use. An image of the QR code (linking to 192.168.0.195) can be seen below in Figure 4.

Figure 4: QR Code

12.3.10 HTTP Server Operation

Upon power up, the server first sets up the controllers (see Controller Setup section for more information on this process). Once the controller setup is completed the server alternates between setting up the network connection and monitoring the controllers. This is done so that if a network connection cannot be established, data can still be gathered and cycle information can still be recorded by the server MCU (See network connection and/or Server-Controller Communication for more information). Once the network is setup and established the server switches between monitoring HTTP port 80, checking for an alarm signal from the real time clock (to signal initiate a data save and shift at midnight) and gathering updated machine status information from the controllers (approximately every five minutes, see Server-Controller Communication section for more information).

12.3.11 Real Time Clock

The real time clock (RTC) chip is used by the server to determine when to save and shift the saved cycle data. The real time clock is able to keep very accurate time and date information both when being powered by the server, and when running on its own backup battery (when the server is powered off). The Laundry Now server uses the date information from the RTC to determine what day of the year it is, so corresponding cycle data is matched with the correct day of the year (used in the calculation of number of cycles per month and per week, see SD card section for information on data formatting).

On startup of the server, a command is sent to the real time clock over I2C to first clear any active alarms (in case one was triggered while the server was powered down). Then a command is sent to reset the alarm to trigger at midnight. When midnight occurs, a square wave is output from the RTC chip that is sent through a 47uF capacitor, which results in a steady “high” signal sent to a digital I/O pin on the WF32. The server is constantly monitoring this pin, when it goes “high” it triggers a program on the server to shift all the data in the RAM on the PIC by one day, updates the day, week, and month cycle calculations then backs all of this information up on the SD card (see SD card section for data formatting on the SD card). The alarm on the RTC is then cleared and reset to trigger at the next interval (midnight of the next day).

The real time clock and 47uF capacitor are mounted on top of the server PCB board next to the controller connectors. The output square wave and capacitor are connected to Pin 35/Port RD11 on the WF32 (see schematic for wiring details).

12.3.12 Day of the year

The RTC is used to calculate the day of year which is used in keeping track of which cycle information corresponds with which day. The RTC provides the day of the month and month of the year. From this information the server can calculate the day of the year (out of 365).

12.3.13 SD Card

The SD card module is mounted on the WF32 board and is accessed by the server over SPI. The WF32 implements a “bit-banged” SPI software program in order to access the SD card module, and is not connected to the dedicated SPI module built into the PIC32 MCU. Due to this type of hardware implementation, Digilent Arduino style programming and libraries are used to access the SD card module.

The SD card is used to backup information in case of power loss. The intervals of data backup depends on the file and specific information. See Data Format Cycle Information section for more details on backup intervals.

12.3.14 Data Format Cycle Information

The Laundry Now server uses two data files on the SD card in order to backup cycle information for the last day, week, and month.

The first file stores the cycle information for the last day, and is kept separate because it is updated every five minutes along with the collection of data from the controllers. The file is “LAUN_DAY.txt” and the format begins with four pound signs followed by both a carriage return and new line character. This first line is how the server determines if it is a valid file. If the four pound signs followed by the “\r\n” character are not present or contain any other combination of characters the system determines the file is not valid, deletes the file, and rewrites it with a valid template. If no “LAUN_DAY.txt” file is present then the program creates a blank template file to be updated later.

The second file stores the cycle data over the last month. This file is “LAUN_NOW.txt” and is the largest file stored on the SD card. The file starts with the same four pound signs followed by three integers that correspond with the day of the year (see RTC section for how this is determined). The next line down begins the actual cycle data information. The first column corresponds with the day of the month, the second column corresponds with the controller number. The following eight columns correspond with the individual sensor cycle data. See Figure 5 below for a visual representation of the file formatting.

```
####Last Known Day
Day Number - Controller Number - S1 - S2 - S3 - S4 - S5 - S6 - S7 - S8
```

```
####091
000 001 999 000 000 000 000 000 000 000
000 002 000 000 000 000 000 000 000 000
000 003 000 000 000 000 000 000 000 000
000 004 000 000 000 000 000 000 000 000
001 001 000 000 000 000 000 000 000 000
001 002 000 000 000 000 000 000 000 000
001 003 000 000 000 000 000 000 000 000
001 004 000 000 000 000 000 000 000 000
002 001 000 000 000 000 000 000 000 000
002 002 000 000 000 000 000 000 000 000
002 003 000 000 000 000 000 000 000 000
002 004 000 000 000 000 000 000 000 000
003 001 000 000 000 000 000 000 000 000
003 002 000 000 000 000 000 000 000 000
003 003 000 000 000 000 000 000 000 000
003 004 000 000 000 000 000 000 000 000
004 001 000 000 000 000 000 000 000 000
004 002 000 000 000 000 000 000 000 000
004 003 000 000 000 000 000 000 000 000
004 004 000 000 000 000 000 000 000 000
005 001 000 000 000 000 000 000 000 000
005 002 000 000 000 000 000 000 000 000
005 003 000 000 000 000 000 000 000 000
005 004 000 000 000 000 000 000 000 000
006 001 000 000 000 000 000 000 000 000
006 002 000 000 000 000 000 000 000 000
006 003 000 000 000 000 000 000 000 000
006 004 000 000 000 000 000 000 000 000
007 001 000 000 000 000 000 000 000 000
007 002 000 000 000 000 000 000 000 000
007 003 000 000 000 000 000 000 000 000
007 004 000 000 000 000 000 000 000 000
008 001 000 000 000 000 000 000 000 000
008 002 000 000 000 000 000 000 000 000
008 003 000 000 000 000 000 000 000 000
```

Figure 5: LAUN_NOW.txt File formatting

12.3.15 Data Import

The data that is backed up on the SD card is automatically imported when the sever powers on. First the daily data is loaded into the RAM on the PIC32 from the SD card

(from the file “LAUN_DAY.txt”), then the data from the last month is loaded into the RAM (from the file “LAUN_NOW.txt”). The last known day (stored on the first line in the file) is loaded, and checked against the current day calculated from the date information provided by the real time clock. If the day of the year matches then the sever has not been powered down for more than a day and the current data is matched properly and the 24 Hour, Last Week, and Last Month calculations are performed and updated for the website. If the last known day and current day does not match, then a data shift is required to match the data with the correct date. The number of days the server was powered down is calculated, and the data is shifted that many days. If there is any data from the daily file, it is transferred and shifted into the last month of known values and shifted as needed. The 24 Hour, Last Week, and Last Month calculations are performed and saved for the website. The file is then backed up and updated to reflect the data shift and normal operation of the server continues.

12.3.16 Server – Controller Communication

The server and controller communicate over the UART protocol that is available on both the PIC32 (server) and PIC18 (controller node) microcontrollers. The server is the only device to initiate a communication sequence over UART, and it does so to request data (see controller setup and update data sections for more information) from the controllers. The Laundry Now system uses the UART configured to a baud rate of 1200 bits per second and utilizes chip select control lines running to each controller to select which controller should be communicating. There can be a maximum of four controllers, each with a maximum of 8 sensors, attached to the server. The sever requests updated information from the controllers every five minutes.

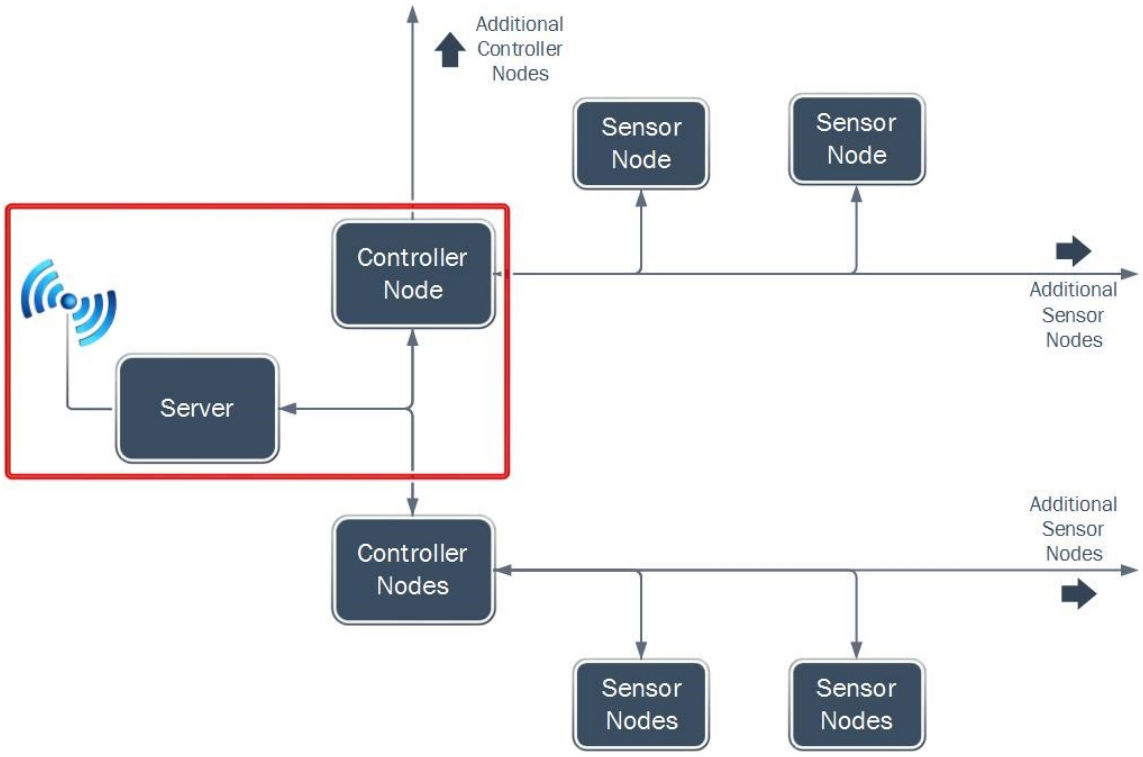


Figure 6: Microcontroller Diagram

12.3.17 Connectors

The server and controller use RJ45 Ethernet connectors to communicate. Straight through Ethernet cabling is required. A maximum of 10ft. is supported (see Testing section in Appendix for test results). Pinout connection schematic is shown in the Figure 7 below.

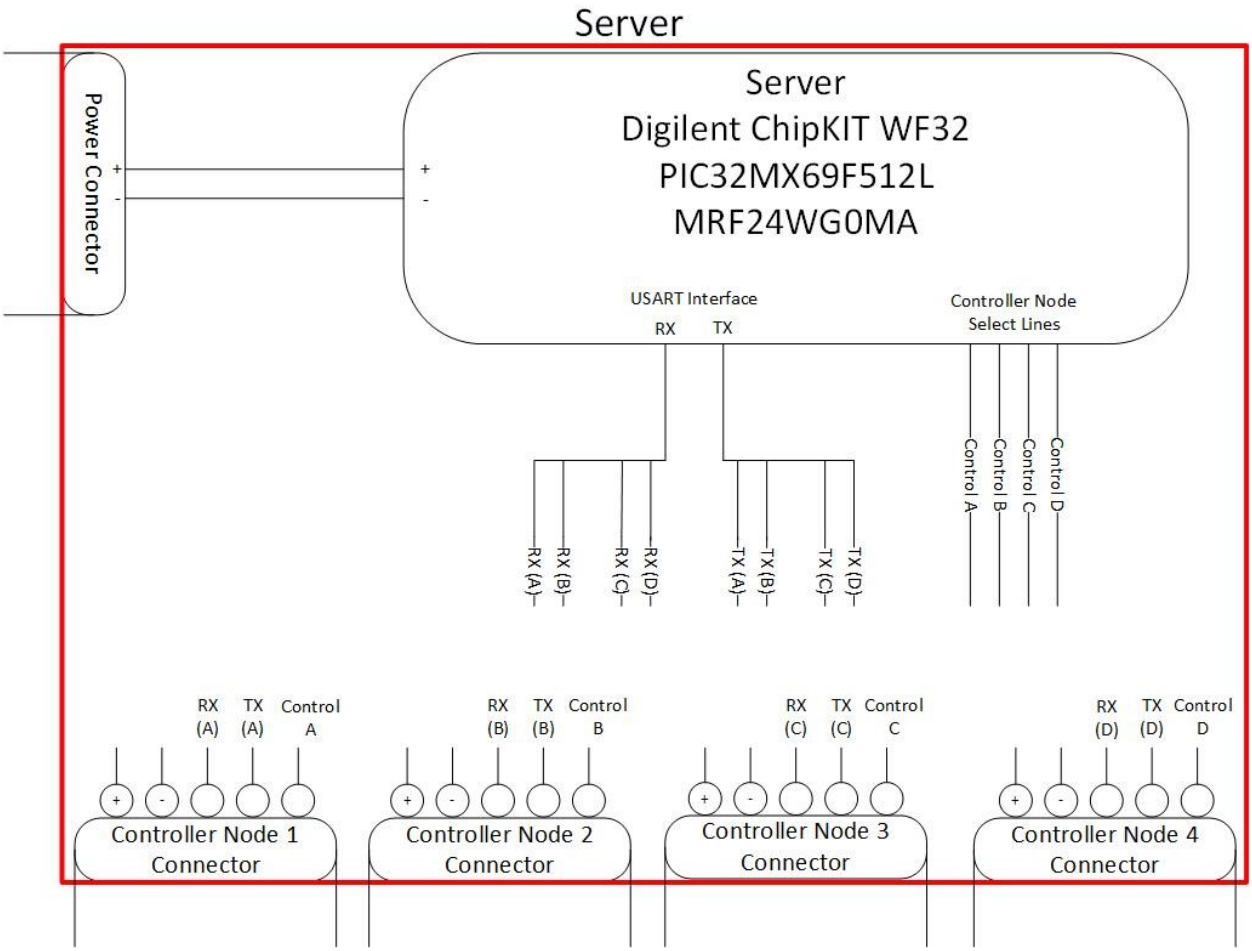


Figure 7: Sever Wiring Layout

12.3.18 Server UART

The server uses the UART module on the PIC32 which corresponds with TX pin 40 and RX pin 39 data lines along with 4 chip select lines (one to each controller). Controller chip select 1 corresponds with pin number 30, controller chip select 2 corresponds with pint number 31, controller chip select 3 corresponds with pin number 32, and controller chip select 4 corresponds with pin number 33. The chip select line is brought low when the server is ready to transmit data, and bits are transmitted to the controller (see Server Controller Data Bits for a list of codes). The chip select line is held low until all communication has been completed between the two MCUs.

12.3.19 Setup Controllers

When the server is first powered on, it must first connect with the controllers to determine how many sensors and controllers are connected to the server. This is the first operation that is performed. First, the server sends a check signal to the controller and waits for a response. The server will send 25 requests before it locks in a time-out condition and determines the controller is not active. If the server received the OK signal from the controller, it continues with setting up the controller and requests the number of sensors. The controller responds with the number of sensors connected, and the server saves this information for building the website later. This process is repeated for each controller; see Setup Controllers Flowchart for a visual representation of this process. For description of the codes being sent see the Server – Controller Data Bits section.

12.3.20 Update Data

Approximately every five minutes the sever requests the data bytes from the controllers connected. This information holds the last known machine status information that was determined by the controller. There are three data bytes (see Server – Controller Data Bits section for a more in depth description of the bits) requested during this operation. The first data byte represents the “active” machines. Machines that are considered active are marked with a logic “1” in the corresponding bit number (see diagram in the Server – Controller Data Bits section for a visual representation of this). The second byte requested is the machines that are in an “inactive” state. The bit number corresponding to the machine number is marked with a logic “1” if the machine is inactive. The third byte requested is the “error” byte. If a sensor stops responding to the controller or the controller cannot determine if a machine is running, the controller can mark the sensor in “error” in the corresponding bit number in byte 3 to display an error on the website. It is important to note the server also error checks the bytes, if a particular machine is marked as “active” and “inactive” or neither “active” nor “inactive” then the website will display an error.

When a machine changes from “active” to “inactive” the website will display that the corresponding machine has “Just Finished.” This is to let users know that though the machine is no longer considered “active” at that time, but there is a high chance the

machine is still occupied. The website displays this status until the next update, at which point the website will then return to reporting the status that the controller has determined.

12.3.21 Server – Controller Data Bits

The controller and server use custom specified coded bytes to transmit information between each other. The list of codes is below.

The following codes are specifically used for setting up the controllers when the server is first powered on.

0b11110000 (0xF0): Server to Controller: Check Signal

Check signal sent to controller to determine if it is active. Part of the setup controller's function. If the server receives the correct response code, the server notes that the controller is active and available to receive commands.

0b10000000 (0x80): Controller to Server: OK Signal

OK signal sent in response to the check signal from the server. Part of the setup controller's function. This tells the server that it is active and available to receive commands.

0b11100000 (0xE0): Server to Controller: Number of Sensors Request

Controller-Sensor request signal, part of the setup controller's function. This code is sent from the server to controller asking for the number of sensors connected to the controller. The number returned by the controller is stored on the server for building the webpage.

The following codes are data request bytes sent from the server to the controllers to gather the latest status information on the machines. There are three bytes total for this transmission. The codes and their corresponding meanings are listed below as well as a mapping diagram to specify what the individual bits correspond to.

0b10000001 (0x81): Server to Controller: Data Byte 1: Active

The first of three bytes,this byte contains information corresponding to machines that are in an “active” state.

0b10000011 (0x83): Server to Controller: Data Byte 2: Inactive

The second of three data bytes, this byte contains information corresponding to machines that are in an “inactive” state.

0b10000111 (0x87): Server to Controller: Data Byte 3: Error

The final of three data bytes, this byte contains information corresponding to machines that are in a state or error.

0b100001111 (0x8F): Server to Controller: Data Byte 4: Unused.

Currently byte 4 is not used by the server. The code structure is implemented in both the controller and server but it is unused. Utilization of this byte is mentioned as part of the future work for door open/closing status information.

Note: If any machine is marked in a state of “error” from the controller (data byte 3), or a particular machine has multiple “status” bytes set (see Data Bit diagram for the mapping of data bits to machines), an error will be displayed on the website.

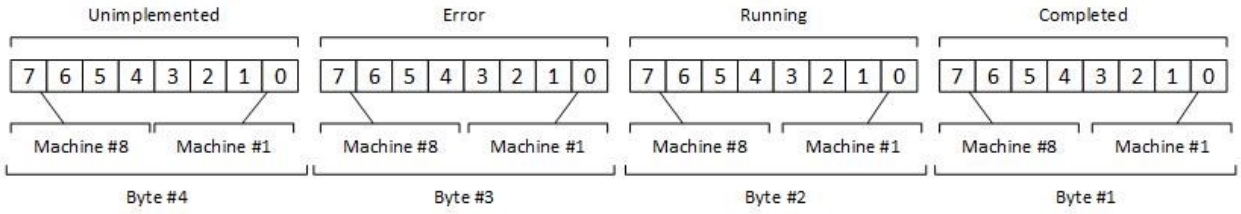


Figure 8: Mapping Diagram of Bits

12.3.22 Controller Node Components

The controller node is a Microchip PIC18F45K20. The controller node is in charge of initializing and monitoring the 8 sensor nodes that are connected to the board in a daisy-chain style connection. There is also a non-inverting Schmitt trigger line buffer used to clean the data signals being transmitted between the PIC MCU and the sensors. The controller node can monitor either washers or dryers, which the user determines by setting the configuration switch on the controller node. When the switch

is set to “low,” the controller is set to dryer mode and changes the algorithm appropriately. When the switch is set to “high,” the system is set to monitor washing machines. Laundry Now does not support the use of different types of machines on a single controller. Each controller must either be all dryers, or all washers. Most laundry rooms are configured in this type of setup due to the hookups required by the machines (water/drains for washers, vents/gas line for dryers).

The controller node board should be placed on the end of the row of machines and can be either mounted on the end machine or the wall with double stick tap. See Figure 9 for the pinout of the controller.

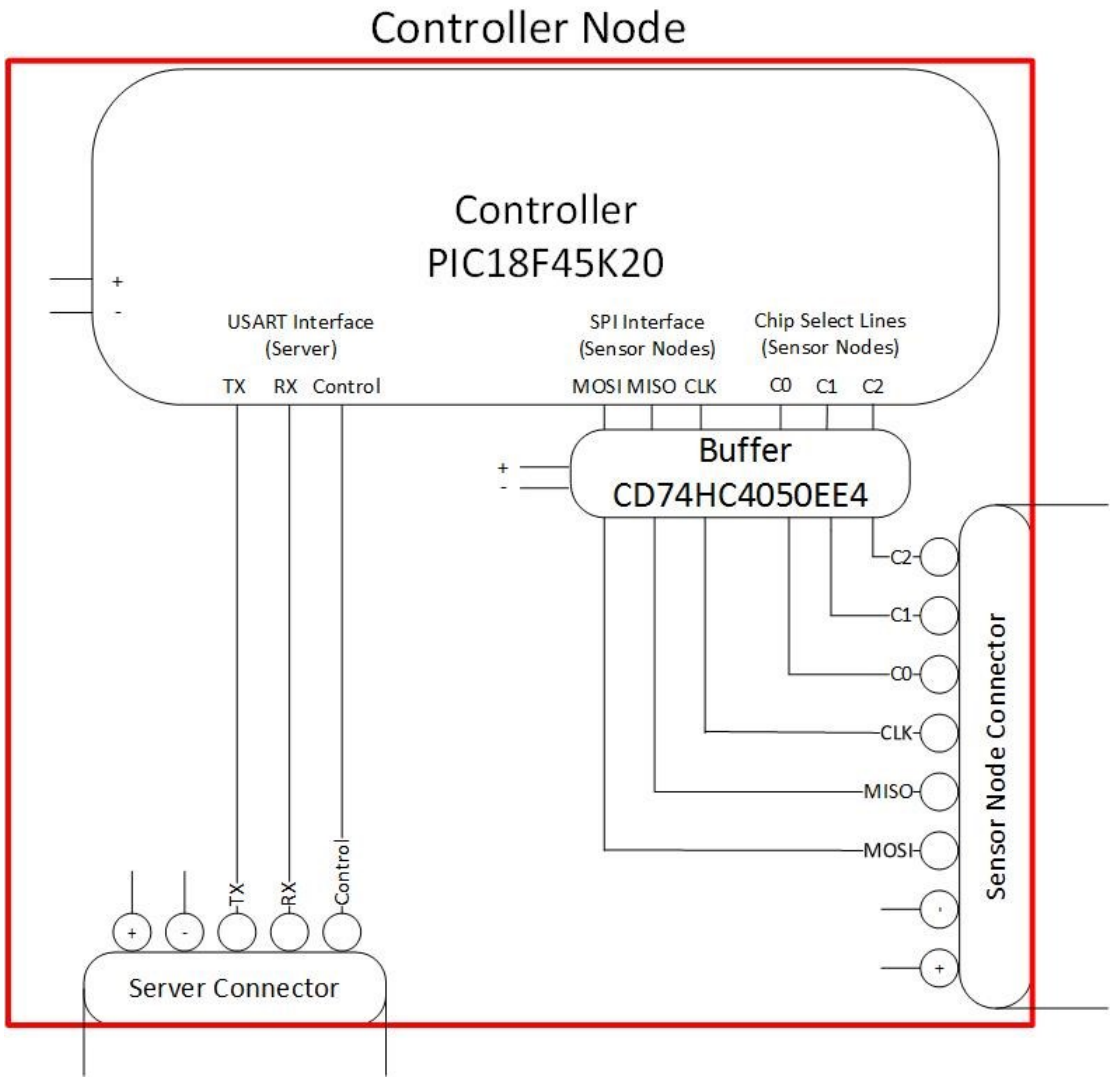


Figure 9: Controller Wiring Diagram

12.3.23 Sensor Node Components

The sensor nodes are designed to be placed on the back of either a washer or dryer and then daisy-chained together with straight though Ethernet cables and RJ45 connectors to their respected controller node. On the sensor node board there is an accelerometer which is used to monitor the movement of the machine. There is a non-inverting Schmitt trigger line buffer that is designed to buffer the communication data lines between the sensor nodes and the controller node. There is a demultiplexing chip which acts as the chip select for the sensor. It demultiplexes three control lines from the controller node and allows for up to 8 different isolated outputs based on the binary value of the control lines. This allows the installer to manually select the address of each individual sensor node and also allows an inexpensive universal design of one common sensor node. It is important to note that every sensor node connected to a controller node must have a separate and independent address or the system will not work properly.

The sensor node board is designed to be mounted on the back of the machine with double stick tape. The sensor itself will be attached directly to the machine using a magnet that will be placed on the back of the sensor. The magnet will stick through the box the sensor node is mounded inside of to allow a direct connection of the sensor to the machine. In testing, it was determined that this was the best means of transferring the machine movement to the sensor. It was also determined that the addition of the magnet on the sensor had no effect on the SPI communication or the operations of the sensor. See Figure 10 for a pinout of the sensor node.

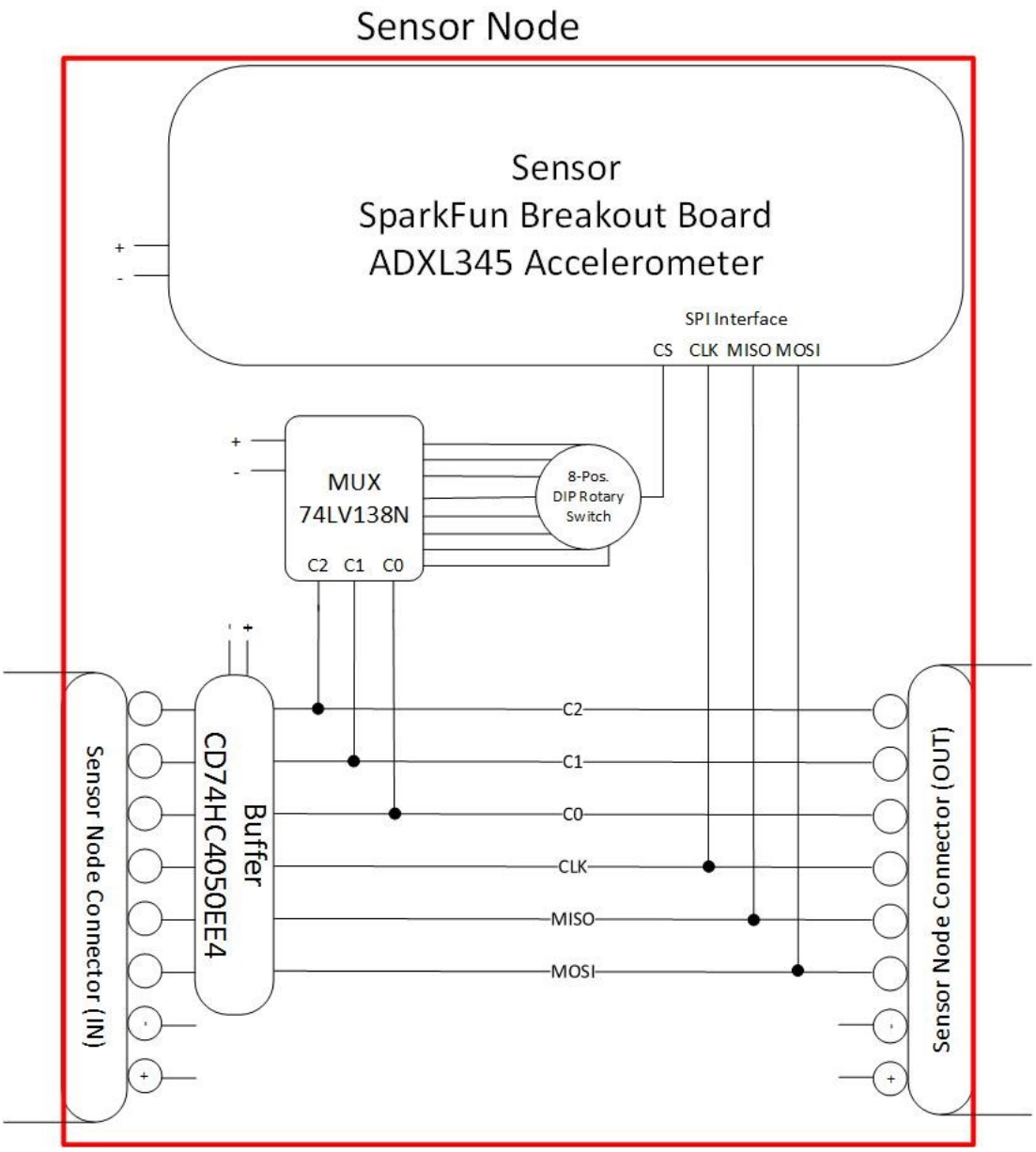


Figure 10: Sensor Wiring Diagram

12.3.24 Series SPI Communication

SPI communication is used to connect the controller node and the sensor nodes. This communication is managed by the controller node in a controller/controlee configuration. The ADXL345 sensor which is implemented on the sensor node can use 3-wire or 4-wire SPI communication but the Laundry Now system uses 4-wire SPI communication in order to reduce the chance of errors being generated on the data

lines. The controller node selects which sensor node it wishes to communicate with by sending a different binary value to the de-multiplexing chip on the sensor node.

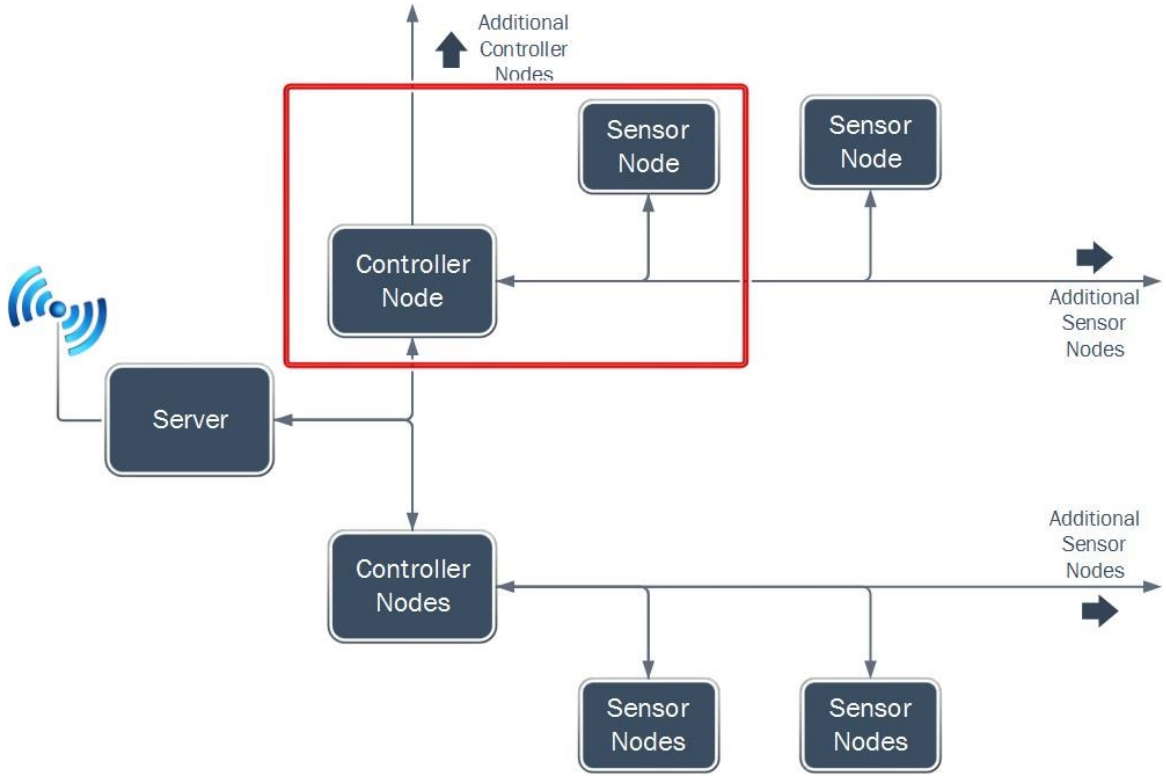


Figure 11: Controller Node to Sensor Node Connection

12.3.25 Overview of Controller Algorithm

The controller node is designed to work with either washing machines or dryers. When the controller node is first turned on it first initializes the PIC18 MCU, SPI MSSP module and the UART communication module. Once this is done the controller will request the device ID from every sensor node to determine the total number of sensors daisy chained together. This value is used by the server to build the webpage. Once it has done this, the controller checks the algorithm select switch to determine whether it will be monitoring washers or dryers. If the switch is set to high then the controller node will enter into the washing machine algorithm. If this switch is set to low then the controller node will enter into the dryer algorithm. This system cannot monitor both washing machines and dryers at the same time. If the user wishes to switch the controller to a different machine type the system must be completely rebooted (Flow Chart 6).

12.3.26 Determination of the Number of Sensors

In order for the controller node to communicate to the sever the number of sensor nodes that are connected, it must send a device ID request right after the PIC MCU and sensor nodes are initialized. The device ID is a known value, and by sending a command of 0b10000000 to the sensor node it can determine if the sensor node is present depending on the returned device ID value. This expected device ID value for the ADXL345 sensors is 0xE5. If the controller receives 11100101 then it increments the number of sensors that are connected to it.

12.3.27 Three Axis Averaging

In order to cut down on errors that may occur the controller node reads the X, Y, Z axis movement values and then averages them with 19 other X,Y, Z points and then averages the three axis's together. This removes the possible spike that could be generated by a sudden movement (door slams, machine being kicked) and also smooths out the movement of the machine so that the controller node can more reliably detect the state of the machine. (Flow chart 9)

12.3.28 Error Checking

Error checking needs to be performed because machines can spend extended periods of time not producing any movement. With the lack of movement and the averaging of the three different axis' over time, there can be long strings of zeros making it difficult to determine if a machine is just not moving, or if a sensor is malfunctioning. In order to tell if sensor is still operating, Laundry Now has a built in error checking function that forces the sensor to respond with a nonzero value. For our error check purposes the controller node will send of the device ID request of 0b10000000, forcing the sensor node to respond with the known ADXL345 device ID of 0xE5. If the controller does not get this value back, then the controller node will flag the sensor as malfunctioning and the server will report this status on the website.

12.3.29 Dryer Algorithm

The controller node will be set to use the dryer algorithm if the switch is in the "low' position. After initializing the sensor nodes, SPI MSSP module, and UART communication module the controller will enter into the dryer algorithm. In the dryer

algorithm the controller node will first take an initial reading by using the three axis averaging. If the averaged movement value crosses over an intensity of 25 then the machine will be flagged a “possible in operation,” but it will not yet officially report as “active” to the server. When the controller node takes the next measurement of the sensor and sees an averaged movement intensity value over 25 the machine will be set into “active” mode and the controller node will report the updated status of the machine to the server upon the next status data transfer. The machine will stay in active mode until the averaged movement value falls below 25 (two samples in a row) or until the sensor does not respond for the device ID correctly. If a sensor enters into the error state the only way that it can exit the error state is if the sensor goes into the active mode (Flow Chart 4 and Flow Chart 5).

12.3.30 Washing Machine Algorithm

The controller node will enter into the washing machine algorithm if the selector switch is set to “high”. When the controller node enters the washing machine algorithm it first requests the averaged movement value of the sensor. If the value is higher than an intensity of 10 then the sensor will be flagged as “possible in operation”. If the sensor is flagged 4 times in a row then the status of the machine will be changed to report as “active” when the server requests the status of the machines. Once it is set into active mode the controller is looking for the sensor to fall below 10. This is to account for the soak cycle of the washing machine. Once it enters into this state the controller is looking for a value of 70 to happen twice, which is for the final spin cycle which generates the most movement. Once the sensor enters into this stage it is looking for a sudden drop in movement to below 10 which indicates that the machine has stopped its cycle and is done. Once this status is reached the controller node will report the machine as “not active”. This algorithm also includes error checking to determine if the sensor is still in communication when it is showing no movement. If the sensor does not report 0xE5 for the device ID check, then the sensor node is reported to be in “error” (Flow Chart 5 and Flow Chart 7)

13 Potential Problems

13.1 False Trigger

Though the washer and dryer algorithm are designed to reduce as many false triggers as possible, it is still possible that certain external conditions could be similar enough to the movement profile of a running washer or dryer and could trigger a false run cycle.

13.2 Machines Too Quiet

Newer machines with enhanced silencing and dampening technologies can be too quiet for the Laundry Now system to detect. Though the algorithms are adaptive, there are still minimum thresholds that have to be met in order to trigger a start condition.

13.3 Wi-Fi Router Not In Range

In some settings, the laundry room may not be within range of a Wi-Fi network. If this is the case, the Wi-Fi network must be moved within range of the Laundry Now server, or the server must be located in a way that it is within range of a Wi-Fi network. Laundry Now does not support direct Ethernet connections.

13.4 Users Not in Range of Wi-Fi

If users are unable to access the network the Laundry Now server is connected to, and port forwarding cannot be setup, then users will not be able to access the webpage that the server generates.

13.5 Unable to Access Back of Machine

If the back of the machine cannot be reached due to permanently mounted machines or other reasons, the system cannot be used effectively. The best location for the sensor to be mounted in order to get the most accurate reading is on the back of the machine. See the testing section for an in depth analysis of the mounting location.

13.6 Loose Connection Due to Continuous Wear

The scope of this project's testing did not include the long term durability of the components. It is possible that over time, the vibrations from the machines can cause damage to the chips, boards, cables, or connecters which may result in the system failing.

14 Failure Analysis & Changes

14.1 Loss of SPI data communication on PCBs

The first parts of the Laundry Now system were tested on breadboards and the initial expected difficulties were tested and eliminated during the early stages of the project. Distance of the SPI interface was tested and it was determined line buffers would be needed in order to clean up the signal to meet the required maximum distance of 10 feet between sensors. Multiple sensors were tested daisy chained together to ensure the extra load from multiple sensors would not corrupt the signal. See the testing section for a more in depth analysis of the tests performed. The portion that could not be tested due to the lack of a breakout board for the RJ45 connectors was the actual Ethernet cable in combination with the RJ45 connectors. When the system was transferred from the breadboard to the PCBs, the data being received from the sensors was erratic and unpredictable. The only sensor observed to be operating correctly was the last physical sensor in the daisy chain (address of the sensors did not matter). Upon careful analysis using the oscilloscope it was determined that the sensors were sending data at the same time and not following the chip select command being sent by the controller.

A side note found later on the data sheet of the ADXL345 states that, when using multiple sensors on the same SPI bus, it is possible that the sensors not selected for communication may unintentionally interpret the clock and data-in activity for an I2C data packet and may try to respond. To correct this an OR logic gate was recommended to be added between the chip select and data-in lines in order to block the data-in line completely when the chip select was not selected. This did not explain

the condition of the last physical sensor reporting correct data, and upon further testing of the OR logic gate correction it was determined this was not the cause of the problem.

After observing the behavior of the data lines on the scope, it was noted that on occasion the data return line from the sensors to the controller would invert itself. The signal should be active low, idle high and during some instances the signal would flip to be active high, idle low. This would cause misinterpretation of data on the controller side.

The exact cause of the signal inversion is unknown, but due to the erratic behavior it was assumed that the extra load of multiple sensors over the large distance on the Ethernet cables was too much for the sensors to handle, and the signal was inverting as a result. Though testing was done on both load and distance, these two factors were not tested at the same time. In addition to this, the Ethernet cables were not tested and it is possible the twisted pair design of the cables caused interference between the data lines causing the signal to invert. On the assumption that load caused the signal inversion, pull up resistors were tested on the data-in and data return lines at the sensor. This forced the signal to stay active low, idle high and normal operation returned to the SPI data lines.

14.2 SD CARD

The original proposal did not include any plans to track data in the event of a power outage. Though the chipKit board has additional SD storage, the only memory that would be utilized was the onboard RAM internal to the PIC32. As the project developed it was determined the system should be able to track usage data over an entire month, as well as during the event of a power outage to the server. This would be accomplished by saving the data into files stored on the SD card. It was also later determined that the Wi-Fi network information would be loaded onto the server via the SD card.

14.3 WPS (WI-FI PROTECTED SETUP)

In the initial proposal for the project it was decided that in order to connect to the Wi-Fi network the server would use push-button Wi-Fi Protected Setup (WPS). With push-

button WPS, the user presses the WPS connect button on the device, then presses the WPS connect button on the router (router dependent). This allows both devices to connect without the need to enter a password allowing Laundry Now to not require a user to manually enter the network information. During the initial research phase of the project it was determined that the MRF24WG0MA was able to support push-button WPS setup. As a side note, the chipKit WF32 board was chosen for the added libraries provided by Digilent. When it came time to implement the push button WPS it was discovered that the Digilent libraries did not support this. Since the SD card had already been added to the system at that time it was decided a program would be created to write a file onto the SD card that contains the network information, instead of writing custom libraries to support WPS.

15 ECONOMIC ANALYSIS

This product is designed to compete with LaundryView, which is a similar product. LaundryView is a system that was created by MacGray to work with the washers and dryers that they supply. Mac Gray is a laundry facility service provider which works with some schools and apartment complexes to maintain their laundry faculties. They provide the machines and perform maintenance on the machines. Their LaundryView system is only designed to work with the machines that they install. They currently do not quote a price on the cost per machine and the price of the LaundryView system, but the machines they install are either Maytag or Speed Queen washers and dryers. The average price of a new Maytag or Speed Queen washer is about $1500.00 per unit, and the average for a Maytag or Speed Queen dryer is about $1500.00 per unit. We wanted Laundry Now to be less expensive than LaundryView. Because of this, our product is designed to work with preexisting washers and dryers in a laundry facility so there is no need to purchase new washers and dryers. The parts cost for a fully expanded (32 sensor nodes, 4 controller nodes and 1 server) Laundry Now system is about $525 at the current parts cost. Using the chips directly (not on breakout boards) would bring the cost of Laundry Now lower [1,2,4].

16 Social and Environmental Impact

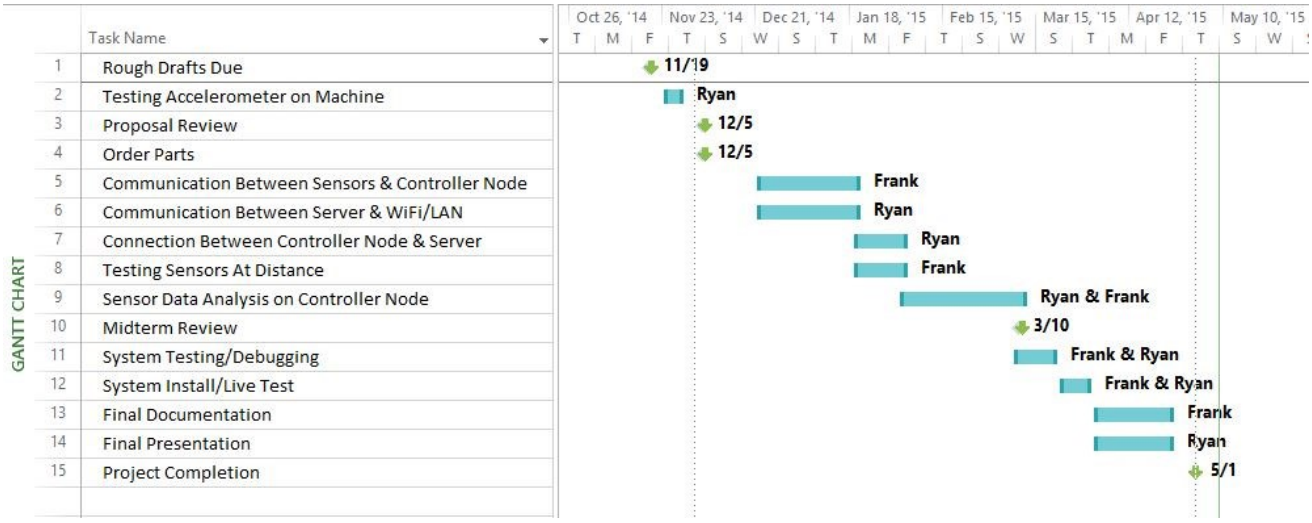
Laundry Now is designed to compete with LaundryView, which is a system that is already on the market and in use by college campuses and apartment complexes. Because of this it will have a similar social and environmental impact to LaundryView which has been greatly successful at reducing the time required by users to do their laundry, and increasing the efficiency of the laundry facilities while being more accessible for facilities where finances may be concerned [2].

17 Timeline and Gantt chart

GANTT CHART

	Task Name	Duration	Start	Finish	Resource Names
1	Rough Drafts Due		Wed 11/19/14		
2	Testing Accelerometer on Machine	5 days	Mon 11/24/14	Fri 11/28/14	Ryan
3	Proposal Review		Fri 12/5/14		
4	Order Parts		Fri 12/5/14		Frank & Ryan
5	Communication Between Sensors & Controller Node	22 days	Mon 12/22/14	Tue 1/20/15	Frank
6	Communication Between Server & WiFi/LAN	22 days	Mon 12/22/14	Tue 1/20/15	Ryan
7	Connection Between Controller Node & Server	11 days	Tue 1/20/15	Tue 2/3/15	Ryan
8	Testing Sensors At Distance	11 days	Tue 1/20/15	Tue 2/3/15	Frank
9	Sensor Data Analysis on Controller Node	27 days	Tue 2/3/15	Wed 3/11/15	Ryan & Frank
10	Midterm Review		Tue 3/10/15		
11	System Testing/Debugging	10 days	Mon 3/9/15	Fri 3/20/15	Frank & Ryan
12	System Install/Live Test	6 days	Mon 3/23/15	Mon 3/30/15	Frank & Ryan
13	Final Documentation	17 days	Thu 4/2/15	Fri 4/24/15	Frank
14	Final Presentation	17 days	Thu 4/2/15	Fri 4/24/15	Ryan
15	Project Completion			Fri 5/1/15	

Initial Time Line

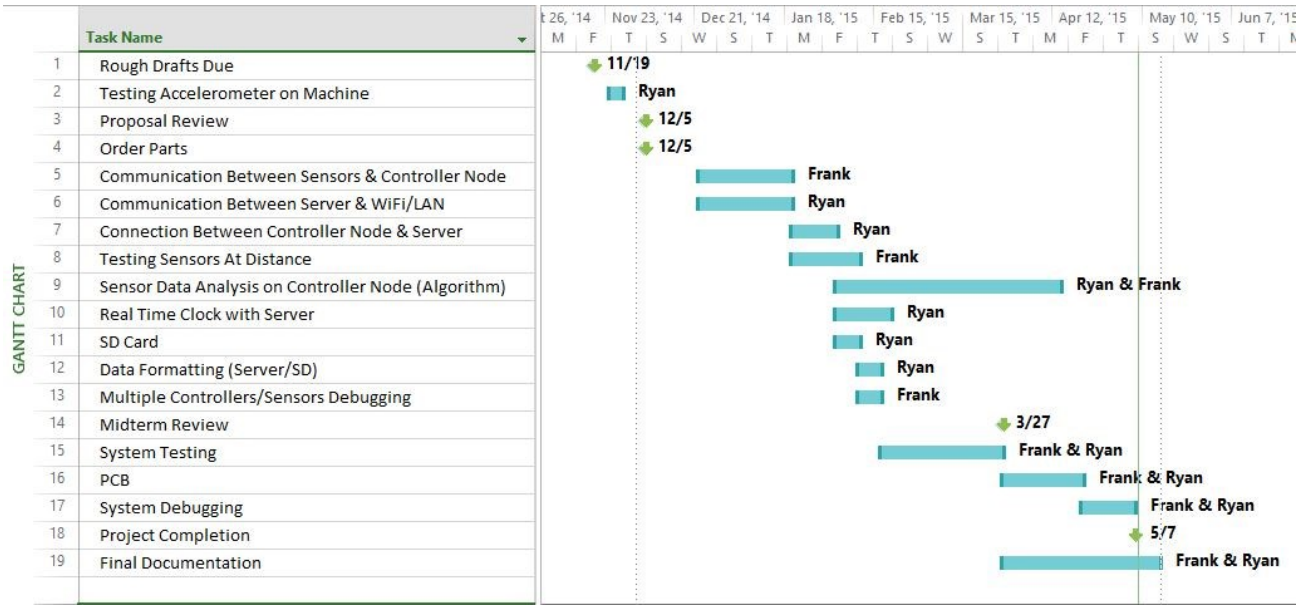


Initial Gantt Chart

GANTT CHART

	Task Name	Duration	Start	Finish	Resource Names
1	Rough Drafts Due		Wed 11/19/14		
2	Testing Accelerometer on Machine	5 days	Mon 11/24/14	Fri 11/28/14	Ryan
3	Proposal Review		Fri 12/5/14		
4	Order Parts		Fri 12/5/14		Frank & Ryan
5	Communication Between Sensors & Controller Node	22 days	Mon 12/22/14	Tue 1/20/15	Frank
6	Communication Between Server & WiFi/LAN	22 days	Mon 12/22/14	Tue 1/20/15	Ryan
7	Connection Between Controller Node & Server	11 days	Tue 1/20/15	Tue 2/3/15	Ryan
8	Testing Sensors At Distance	16 days	Tue 1/20/15	Tue 2/10/15	Frank
9	Sensor Data Analysis on Controller Node (Algorithm)	51 days	Tue 2/3/15	Tue 4/14/15	Ryan & Frank
10	Real Time Clock with Server	14 days	Tue 2/3/15	Fri 2/20/15	Ryan
11	SD Card	6 days	Tue 2/3/15	Tue 2/10/15	Ryan
12	Data Formatting (Server/SD)	6 days	Tue 2/10/15	Tue 2/17/15	Ryan
13	Multiple Controllers/Sensors Debugging	6 days	Tue 2/10/15	Tue 2/17/15	Frank
14	Midterm Review		Fri 3/27/15		
15	System Testing	29 days	Tue 2/17/15	Fri 3/27/15	Frank & Ryan
16	PCB	18 days	Fri 3/27/15	Tue 4/21/15	Frank & Ryan
17	System Debugging	13 days	Tue 4/21/15	Thu 5/7/15	Frank & Ryan
18	Project Completion			Thu 5/7/15	
19	Final Documentation	36 days	Fri 3/27/15	Fri 5/15/15	Frank & Ryan

Final Time Line



Final Gantt Chart

Project Milestones			
Task	Person In charge	Date of Completion	Percent complete
Sensor & Machine Testing	Ryan	12/1/2014	100
Proposal Review			
Ordering Parts	Frank & Ryan	12/5/2015	100
Communication between sensor and MCU	Frank & Ryan	1/20/2015	100
Sever and Wi-Fi communication (User Interface)	Ryan	1/20/2015	100
Connect Node and Sever (Data Packets sent UART)	Ryan	2/3/2015	100
Sensor Data Analysis on Controller Node	Frank & Ryan	3/11/2015	100
Midterm Review		3/27/2015	100
System Testing/ Debugging	Frank & Ryan	3/30/2015	100
System Install/ Live Test	Frank & Ryan	3/30/2015	100
Documentation		4/30/2015	100
Project Completion		5/8/2015	100

18 FUTURE WORK

1. Incorporate door detection into controller algorithm
2. Ability to detect movement on stacked machines by comparison
3. Add a work order request page for user to report broken machines
4. Incorporate graph that displays machine usage data for day, week, and year
5. Server to support more network security types
6. Server supports configurable IP address in the network settings file
7. Implement a pressure sensor which will allow for the determination of when the load of laundry has been removed

19 SUMMARY

Laundry Now will allow users to predetermine if there are washers or dryers that are not in use, or to determine that all of the machines are being used. This product is designed to be placed on any washer and dryer units, including older washers and dryers that are not Wi-Fi enabled. There will be an accelerometer that will be placed on every machine that will then allow the controller node to monitor the washer/dryer movement. The controller node will collect this movement data from the sensor every two minutes. It will analyze the data and update the status of the washer/dryer to one of three states: Available, Running, and Just Finished. It will store this information until the server requests it. Every 5 minutes the server will request the latest machine status from the controller nodes and update the HTML webpage file. When a user checks the website, the server will send the latest HTML file showing the status of the machines. It is important to note that the system will not be able to directly tell if a machine has been emptied. It will monitor the movement of the washer or dryer and make a best determination of whether it is running, has completed a cycle, or is empty.

20 REFERENCE

1) http://www.macgray.com/

   Mac Gray is the inventor and supplier of LaundryView and all of the products needed to run LaundryView. 11/15/2014

2) http://www.campus-solutions.net/index.html

   Campus-solutions is one of the many features that is offered by Mac Gray. This website shows how LaundryView has been implemented in colleges across the country and how it has improved the efficiency of college laundry facilities. 11/15/2014

3) http://www.amazon.com/LG-Electronics-RLM20K-Laundry-Monitoring/dp/B0012OMY2K

   Is the main selling source for LG's Remote Laundry Monitoring System, providing cost along with description of what the product does and some reviews form customers. 11/15/2014

4) http://www.maytag.com/Laundry-1/Laundry_Laundry_Appliances_Washers-3/102120050+4294966015/

   Provides the estimated cost of the Maytag washers and dryers that Mac Gray offers as part of their laundry facility service, and the machines needed to allow for the use of LaundryView. 11/15/2014

5) http://www.nmhc.org/Content.aspx?id=4708

   This link provides the number of residents the live in apartment complexes in the state of California and the whole country. 12/4/2014

6) http://www.analog.com/static/imported-files/data_sheets/ADXL345.pdf

   Analog Devices ADXL345 Data Sheet. 12/1/2014

7) https://www.sparkfun.com/products/9836

   SparkFun Product Page & Details for ADXL345 Breakout Board. 12/1/2014

8) http://ww1.microchip.com/downloads/en/DeviceDoc/41303G.pdf

   Microchip PIC18F45K20 Data Sheet. 12/1/2014

9) http://ww1.microchip.com/downloads/en/DeviceDoc/61156H.pdf

   Microchip PIC32MX69F512L Data Sheet. 12/1/2014

10) http://ww1.microchip.com/downloads/en/DeviceDoc/70686B.pdf
Microchip MRF24WG0MA Wi-Fi Controller. 12/1/2014
11)https://www.digilentinc.com/Products/Detail.cfm?NavPath=2,892,1193&Prod=CHIPKIT-WF32
Digilent chipKit WF32 Product Page. 12/1/2014

21 APPENDICES

21.1 DETAILED BUDGET

Laundry Now Budget							
Item	**Part No.**	**Purpose**	**Manufacturer**	**Supplier**	**Price**	**Quantity**	**Ext. Price**
Accelerometer	ADXL345	Sensor Package (SPI)	Analog Devices	SparkFun	7.95	4	31.8
Microcontroller	PIC18F45K20	Controller	Microchip	DigiKey	2.89	3	8.67
Microcontroller/WiFi Module	410-273P-KIT	Server	Microchip/Digilent	Digilent	69.99	1	69.99
Demultiplexer	74LV138N	Chip Select	NXP Semiconductors	Mouser	1.53	4	6.12
Wire	T1290-30-ND	Wire, 8 twisted pair	Tensility	Mouser	40	1	40
							156.58

This budget reflects the cost of parts to make the demo unit using breakout boards and pre-assembled parts (1 server, 2 controller nodes, 4 sensor nodes). If Laundry Now was to be implemented as a full scale commercial product, breakout boards and pre-assembled boards would not be needed and the cost for the chips would drop, allowing the overall cost for the system to be cheaper. The budget below shows the cost using current parts (breakout boards/pre-assembled boards) for a fully expanded system (1 server, 4 controller nodes, 32 sensor nodes).

Laundry Now Device Suppliers					
Item	**Part No.**	**Purpose**	**Manufacturer**	**Supplier**	**Link**
Accelerometer	ADXL345	Sensor Package (SPI)	Analog Devices	SparkFun	https://www.sparkfun.com/products/9836
Microcontroller	PIC18F45K20	Controller	Microchip	DigiKey	http://www.digikey.com/product-detail/en/PIC18F45K20-I%2FP/PIC18F45K20-I%2FP-ND/1228477
Microcontroller/WiFi Module	410-273P-KIT	Server	Microchip/Digilent	Digilent	http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,892,1193&Prod=CHIPKIT-WF32
Demultiplexer	74LV138N	Chip Select	NXP Semiconductors	Mouser	http://www.mouser.com/ProductDetail/NXP-Semiconductors/74LV138N112/?qs=sGAEpiMZZMtxONTBFIcRfsgCRr8SIzJ%2fBpNkyO2p2jo%3d
Wire	T1290-30-ND	Wire	Tensity	Mouser	http://www.digikey.com/product-detail/en/30-00392/T1290-30-ND/5023179

21.2 TEST AND MEASUREMENTS

21.2.1 Initial Proof of Concept

In order to be sure the microcontrollers could properly collect valid movement data from the machines we tested a similar accelerometer. We used an Arduino Uno microcontroller with a simple script to monitor the machine, collect the data, and transmit it back to a laptop where the data points could be analyzed. Using MATLAB, we plotted the data points across the clock cycles to see if the vibrations created by the machine could be easily analyzed. We found that we could easily see the movement of the machine when it was running and the quick spikes in movement when the door was opened and closed. This data allowed us to do some simple processing on the controller node MCU to determine if a machine was in use or available. The sensor that was used was a Parallax 2-axis MEMS digital accelerometer that provided an output similar to what the Analog Devices 3-axis MEMS accelerometer will output. Figure 12 is a screenshot taken during the data collection process and Figure 13 is the graph representation of the collected data.

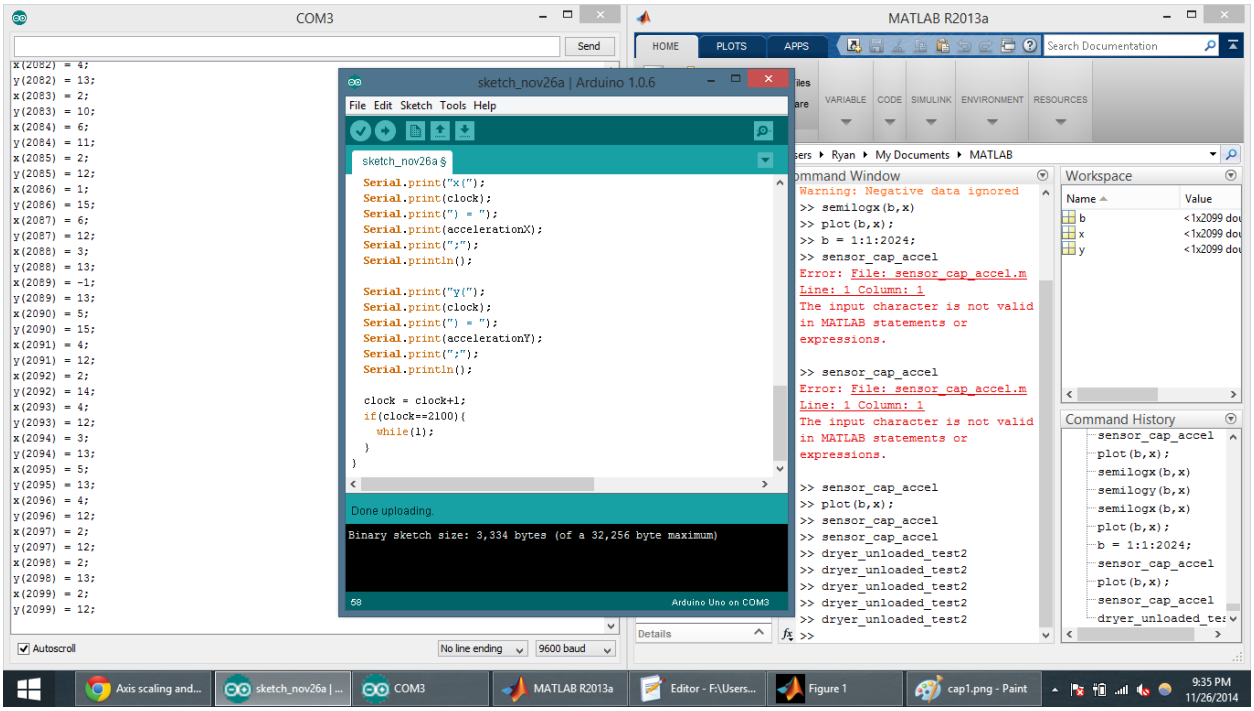


Figure 12: Generating of Test Information

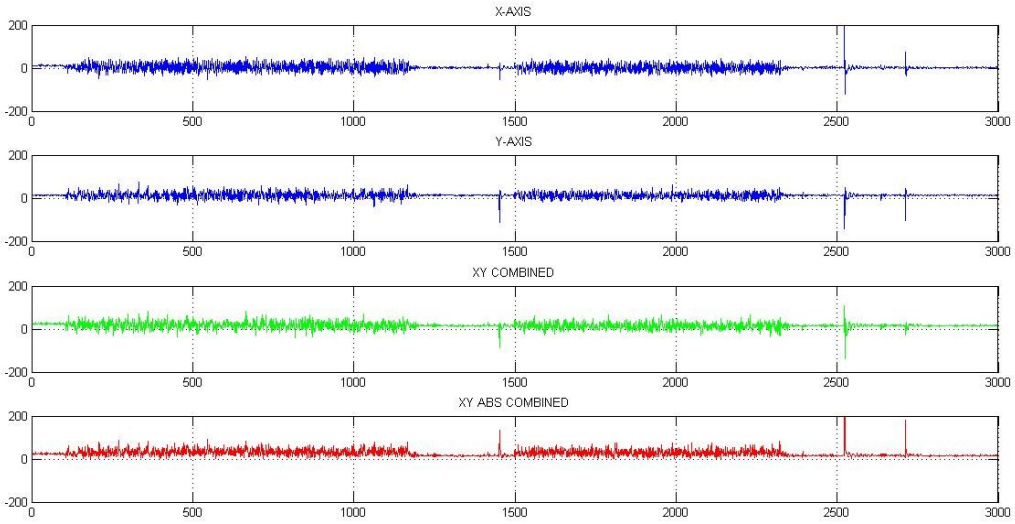


Figure 13: Testing Graphs

The first two graphs are the raw data output of the X and Y axis of the accelerometer. The third graph is one method of analysis to emphasize the spikes by summing the axes. We found that while this helped amplify certain parts of the data, it cancelled out in other cases. In order to get around this, we added the absolute value of the two axes. This produces a much better output that can easily be analyzed by the controller node.

21.2.2 Bench Testing

Laundry Now Bench Testing			
Test	Other Test Notes	Test Conclusions/Results	Graph
Sensor Commutation Testing	Test communication possibilities with SPI to a single sensor.		See Figure
- Device ID		Device ID register request was sent and then Device ID was received as 0xE5	14
- Status Register		The Status register was sent and it was received. This was also to test if the sensor can pick up movement and the sensor does pick up movement.	15, 16

Sensor Setup	Testing the sensor setup processes.		See Figure
- Setup Sensor Communication		The SPI setup code was successful.	17, 18, 19
Serial Communication Over Distance (Sensor Node/ Controller Node)	This test was to determine if buffers were needed to have reliable communication between sensors and the controller over distance.		See Figure
1 sensor, 0 feet, no buffer		Do not need a buffer at zero feet	20
1 sensor, 10 feet, no buffer		Errors start to occur at 10 feet	21, 22, 23
8 sensors, 0 feet, buffer		Buffers added to the signal	24
delay from 8 buffers		Delay of .14us	25, 26
8 sensors, 10 feet, buffers		It works with 80 feet of wire with buffers added ever 10 feet	27
Parallel Communication over Distance (Sever/ Controller Node)	This test was to determine if buffers were needed to have reliable communication between controllers and the server over distance.		See Figure
4 controllers, 0 feet, no buffer		Do not need a buffer at zero feet	28,29
4 controllers, 10 feet, no buffer		Buffers are not needed for reliable communication over distant.	30, 31, 32, 33, 34

21.2.2.1 SPI Communication Test (Controller Node/ Sensor Node)

It is possible to communicate with a sensor via SPI communication between the sensor and the controller. This was tested by checking for the Device ID by sending 0x10 to the sensor. If it was done correctly then the controller should receive 0xE5 as the Device ID. Then the status register was requested to check if there is detectable movement on the sensor and will it transmit this movement over SPI.

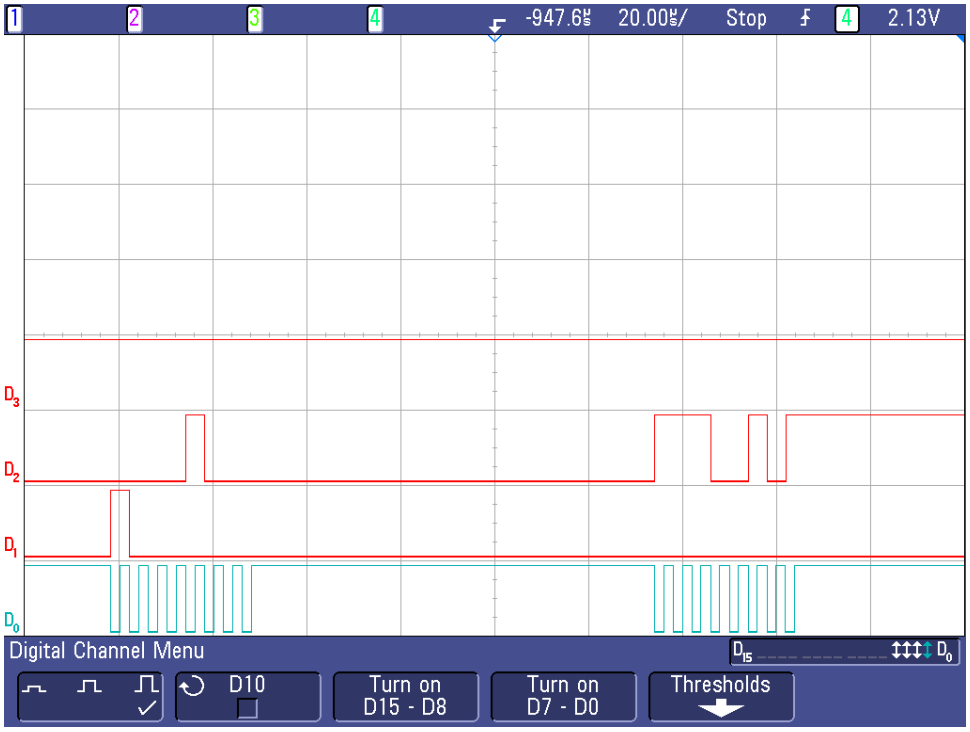


Figure 14: Device ID Request and Transmit

This shows communication back and forth with the sensor and with the controller. D0 shows the SPI clock that is sent by the controller to the sensor, D1 shows the controller output and the sensor input, D2 shows the sensor output and the controller input. D1 shows the request register to the sensor for register 0x00 with the most significant bit indicating it is a read request. Then on D2 shows the sensor response the first series of information is not read by the controller but the second set of information is the response to the request which is 0xE5 or 11100101.

Figure 15: Status Register Request

This scope image shows the request code being sent to the sensor asking for the information stored in the status register. D0 and 1 show the SPI clock, D1 and 2 show the output request from the controller and input of the sensor, D3 and 3 show the output from the sensor and the input from the controller. The first spike on the sensor output is the data ready response and the last two spikes are set when the data ready bit is set.

Figure 16: Status Register Transmission

This shows the response sent from the sensor to the controller. D0 and 1 show the SPI clock, D1 and 2 show the output request from the controller and input from the sensor, D3 and 3 show the output from the sensor and the input of the controller. The spike shows that the activity flag has been set.

21.2.2.2 Sensor Setup Code (Controller Node / Sensor Node)

For the sensors to work they first have to be set up. In order to do this several different registers have to be set up. The registers that need to be set up are the thresh tap, duration, latency, window, thresh act, fifo control, power control, act inact, tap axes, data format, and interrupt map registers.

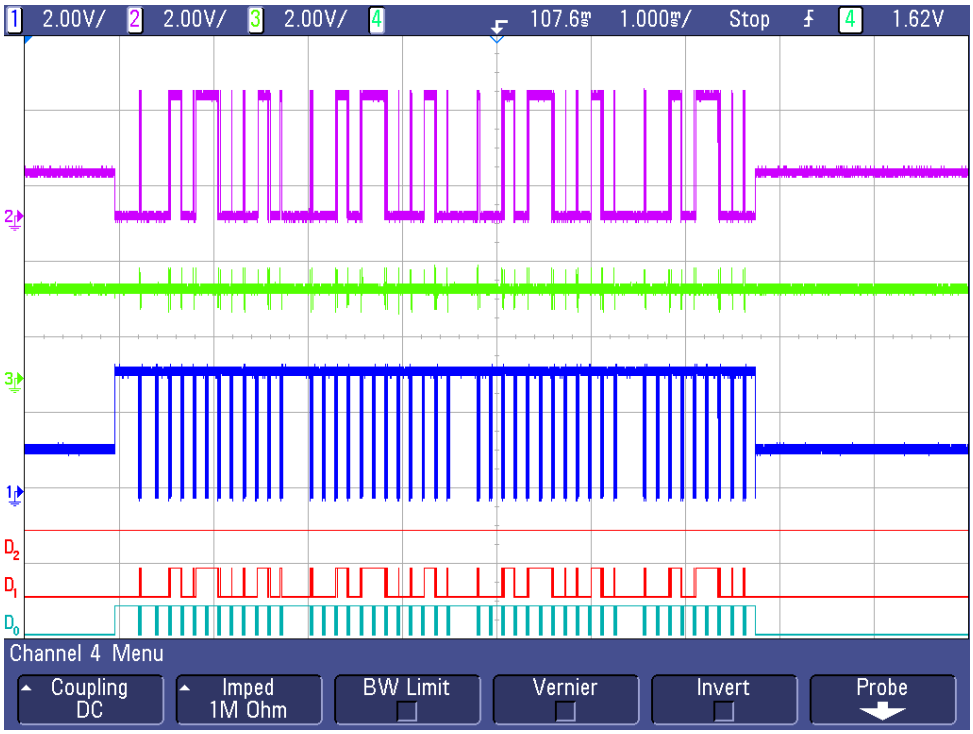


Figure 17: Overview Setup Communication

The image above shows all of the initialization of the sensors being sent from the controller. D0 and 1 show the SPI clock, D1 and 2 show the output request from the controller and input of the sensor, D3 and 3 show the output from the sensor and the input of the controller. D0 and 1 show the SPI clock being sent from the controller; each spike is a total of 8 bits being sent.

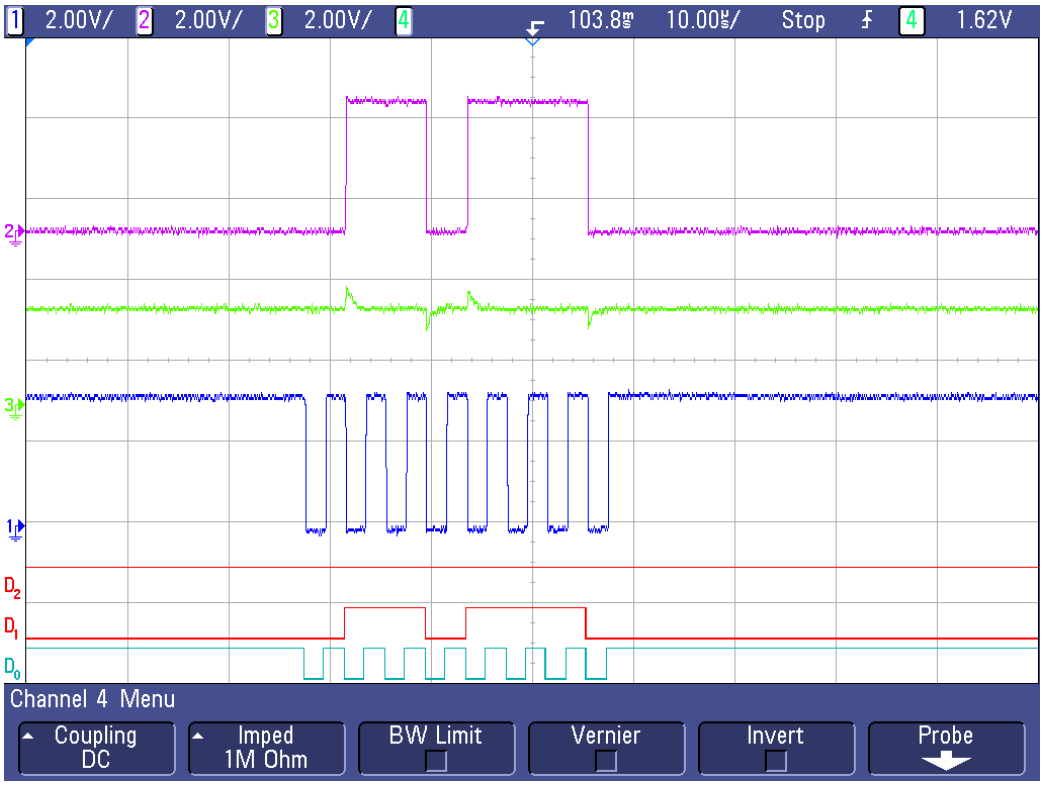


Figure 18: Initial Register Location in Setup

This shows the register location on the sensor (01101110), the most significant bit is for multiple enters being entered, the second most significant bit is indicating a write request, and the rest of the bits are the register location for the int_enabled register. D0 and 1 show the SPI clock, D1 and 2 show the output request from the controller and input of the sensor, D3 and 3 show the output from the sensor and the input of the controller.

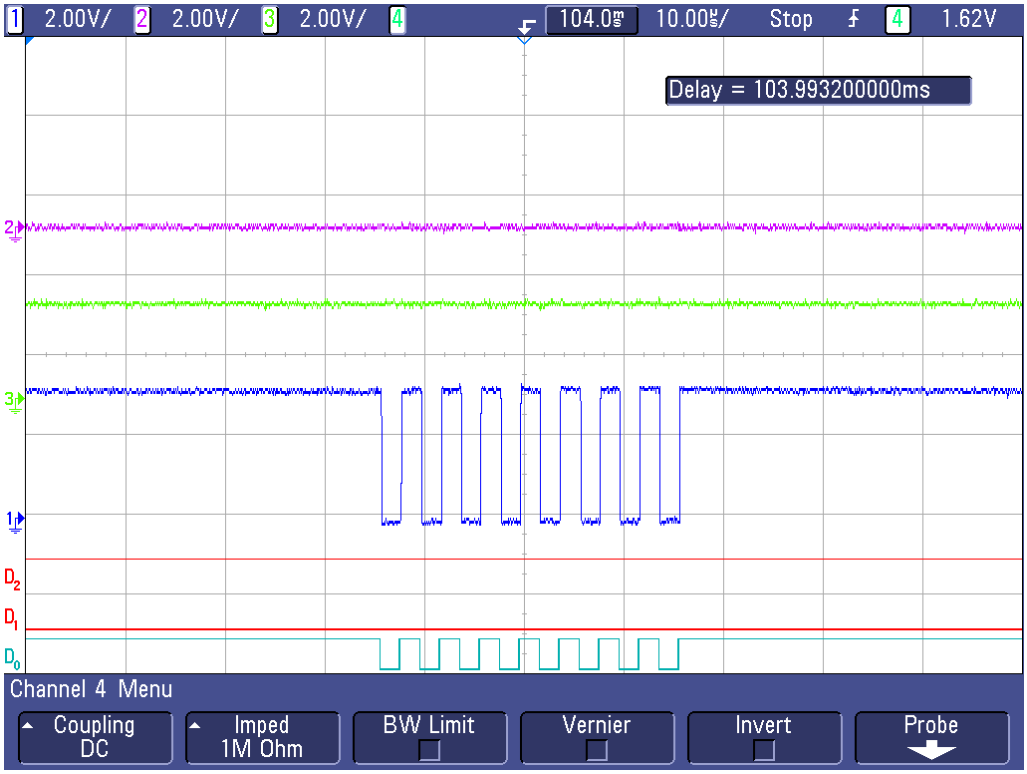


Figure 19: Data being placed in Initial Register

This shows the information being loaded into the int_enabled register or lack of information being loaded (00000000) so that in the initialization process interrupts cannot be triggered. D0 and 1 show the SPI clock, D1 and 2 show the output request from the controller and input of the sensor, D3 and 3 show the output from the sensor and the input of the controller.

21.2.2.3 Serial Communication over Distant (Controller Node/ Sensors Node)

In order for our product to have serial communication from the controller node to the sensor node we will need to determine if there is any data loss caused by both distance and by the load generated by the sensor node. In order to do this we will test communication with the addition of 10 feet of wire between the controller node and the sensor node and then in between every sensor node.

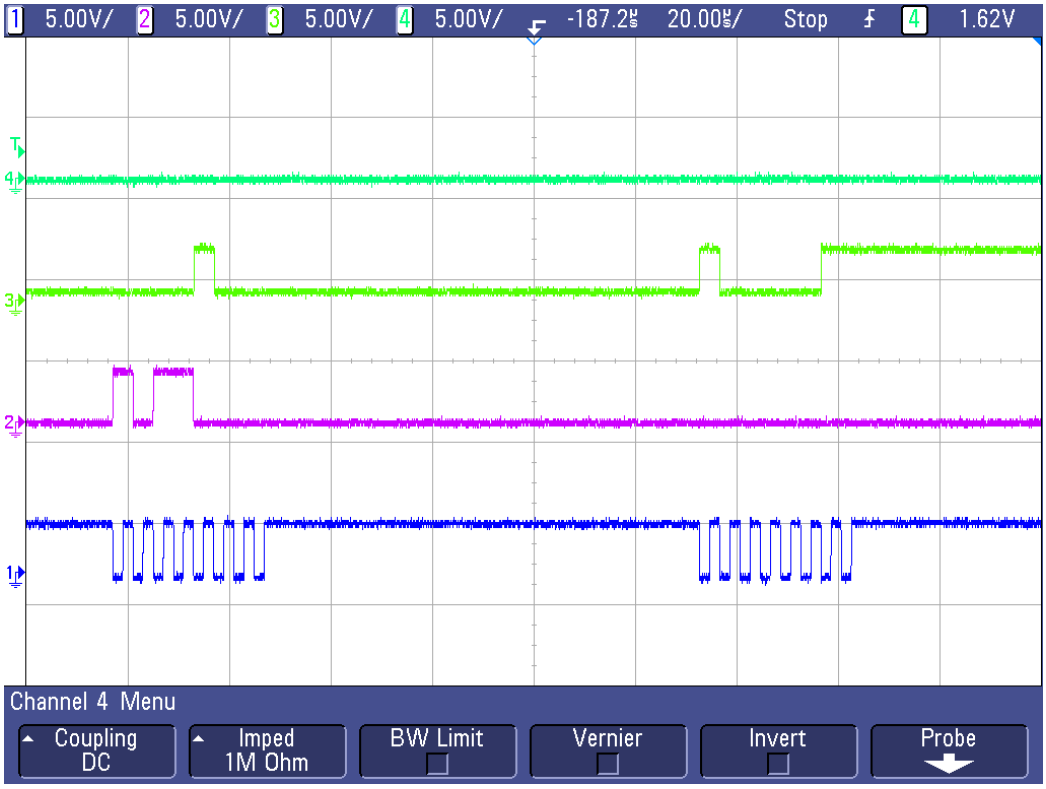


Figure 20: Benchmark, One Sensor, 0 ft Distance

This is a view of the SPI communication without 10 feet of wires and with no buffering of the signal. The signal on 1 is the SPI clock that is sent from the controller to the sensor for communication, 2 is the signal sent from the controller to the sensor to request the information stored in the Int_enabled register, 3 is the signal sent from the sensor to the controller with the values stored in the int_enabled register, and 4 is the control line that is pulled down on the sensor to indicate which sensor is in communication with the controller.

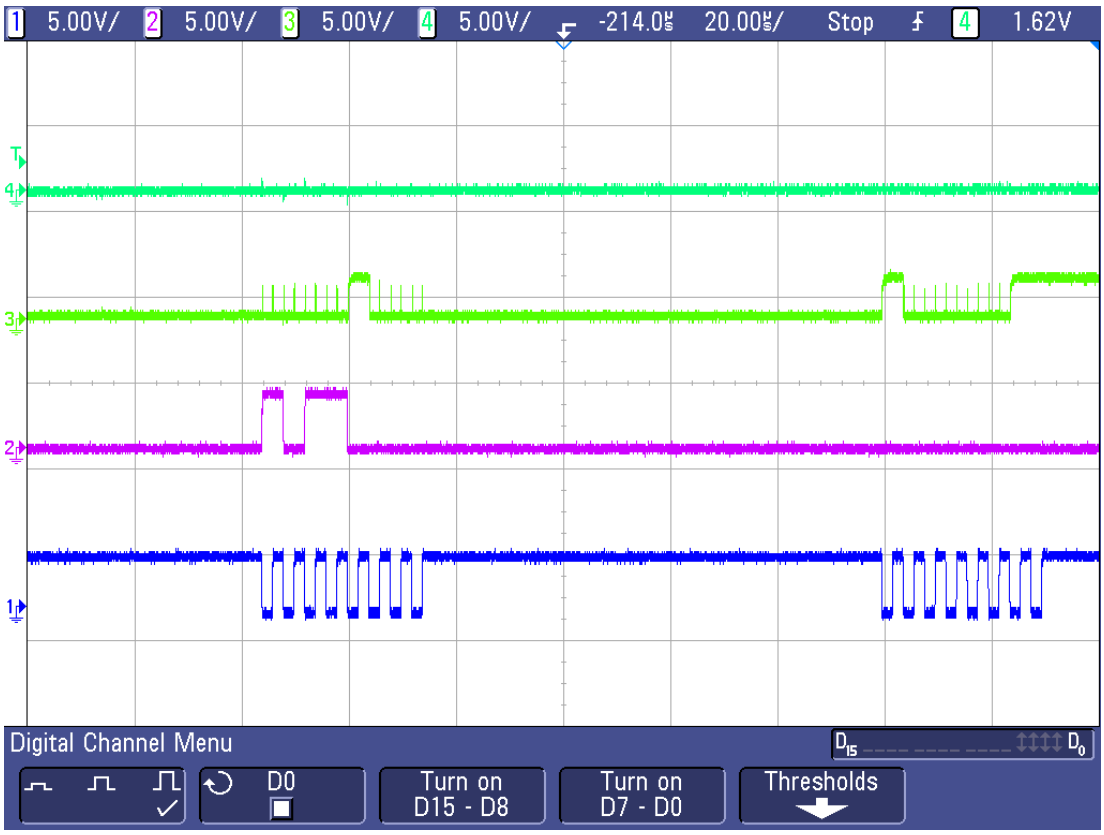


Figure 21: Test, 1 Sensor, 10 ft Distance

This is with 10 feet of wires and with a buffer added. On signal 3 there are some peaks but the controller is still able to read the correct value. The signal on 1 is the SPI clock that is sent from the controller to the sensor for communication, 2 is the signal sent from the controller to the sensor to request the information stored in the int_enabled register, 3 is the signal sent from the sensor to the controller with the values stored in the int_enabled register, and 4 is the control line that is pulled down on the sensor to indicate which sensor is in communication with the controller.

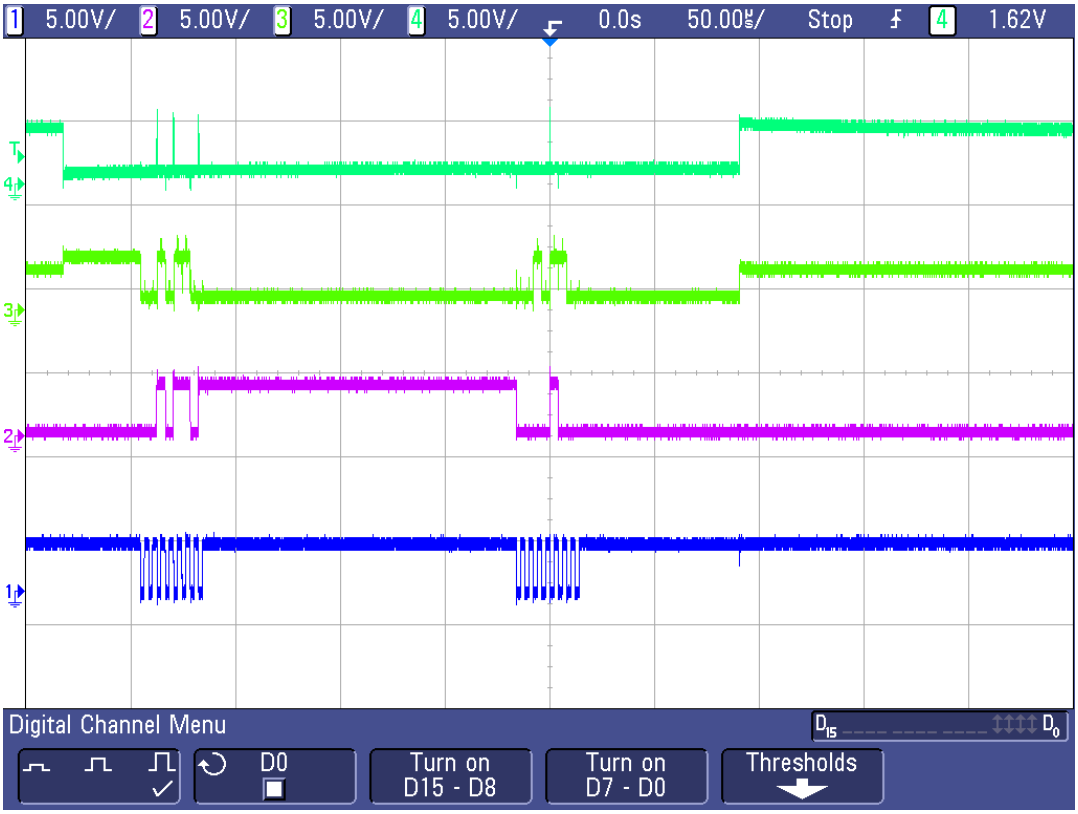


Figure 22: Test, One Sensor, 10 ft Distance, Error Occurring

This is showing the error that occurs on the 10 feet of line without a buffer. The problem that occurs when communicating over distance with no buffer is that the control line will turn off and on randomly, which will cause the sensor to not fully receive the information and to not send the correct information. The signal on 1 is the SPI clock that is sent from the controller to the sensor for communication, 2 is the signal sent from the controller to the sensor to request the information stored in the Int_enabled register, 3 is the signal sent from the sensor to the controller with the values stored in the int_enabled register, and 4 is the control line that is pulled down on the sensor to indicate which sensor is in communication with the controller.

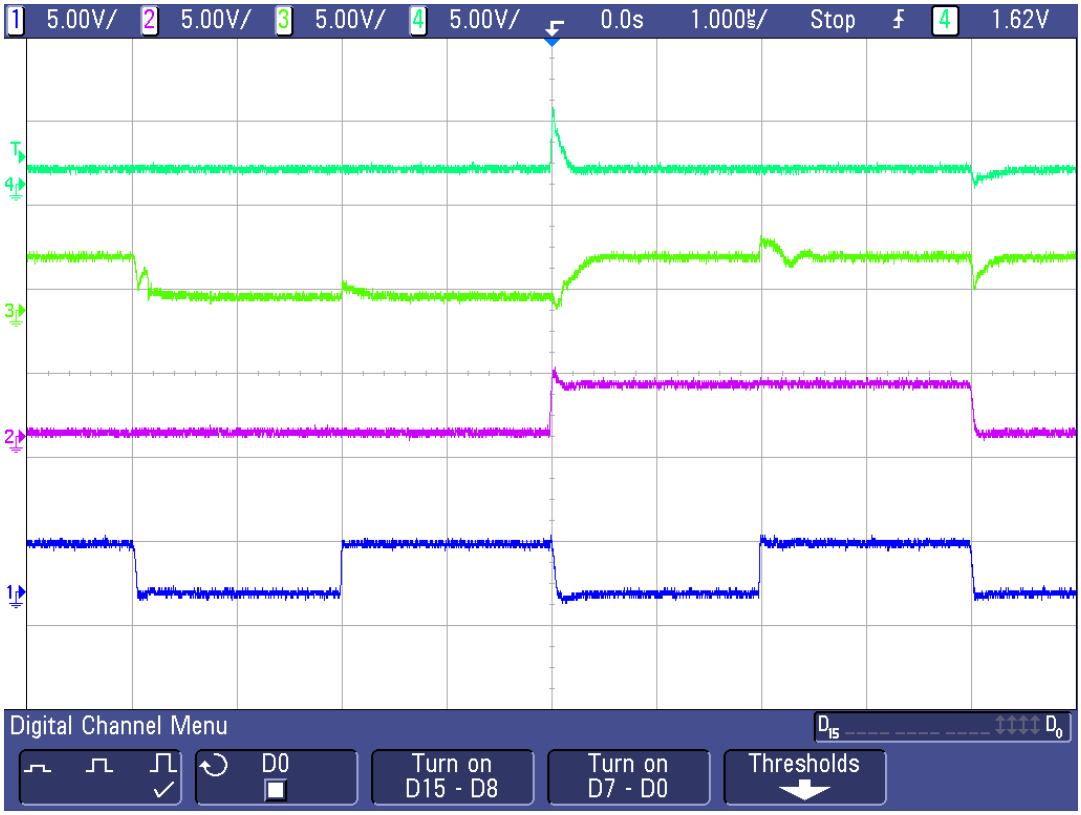


Figure 23: Test, One Sensor, 10 ft Distance, Error Effect on Communication

This is a more zoomed in look at the same error that is occurring in the image above but this is showing on line 4 the error that occurs. When the control line switches to high it causes an error to occur in the rest of the information, and the rest of the communication wires will then sink to the control line error. The signal on 1 is the SPI clock that is sent from the controller to the sensor for communication, 2 is the signal sent from the controller to the sensor to request the information stored in the Int_enabled register, 3 is the signal sent from the sensor to the controller with the values stored in the int_enabled register, and 4 is the control line that is pulled down on the sensor to indicate which sensor is in communication with the controller.

This problem was fixed by adding a non-inverting Schmitt trigger to all of the data communication lines. These allows for the signal to be cleaner when entering the controller node and the sensor node. Without these buffers, communication would not be possible over distance.

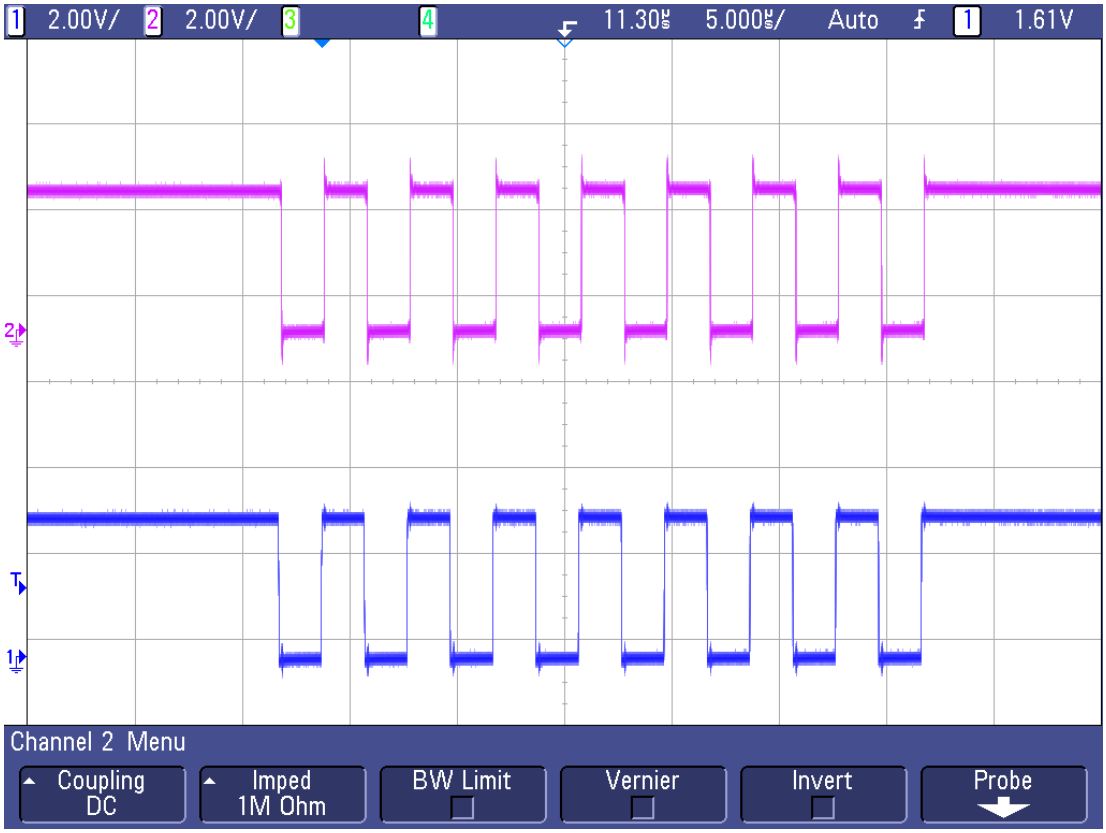


Figure 24: Test, Eight Sensors, Buffers, 0 ft Distance

This shows the SPI clock being transmitted over the buffers without extra wire added. The bottom signal is the signal is the SPI clock being sent from the controller node. And the top signal is the signal after it has passed through 8 buffers.

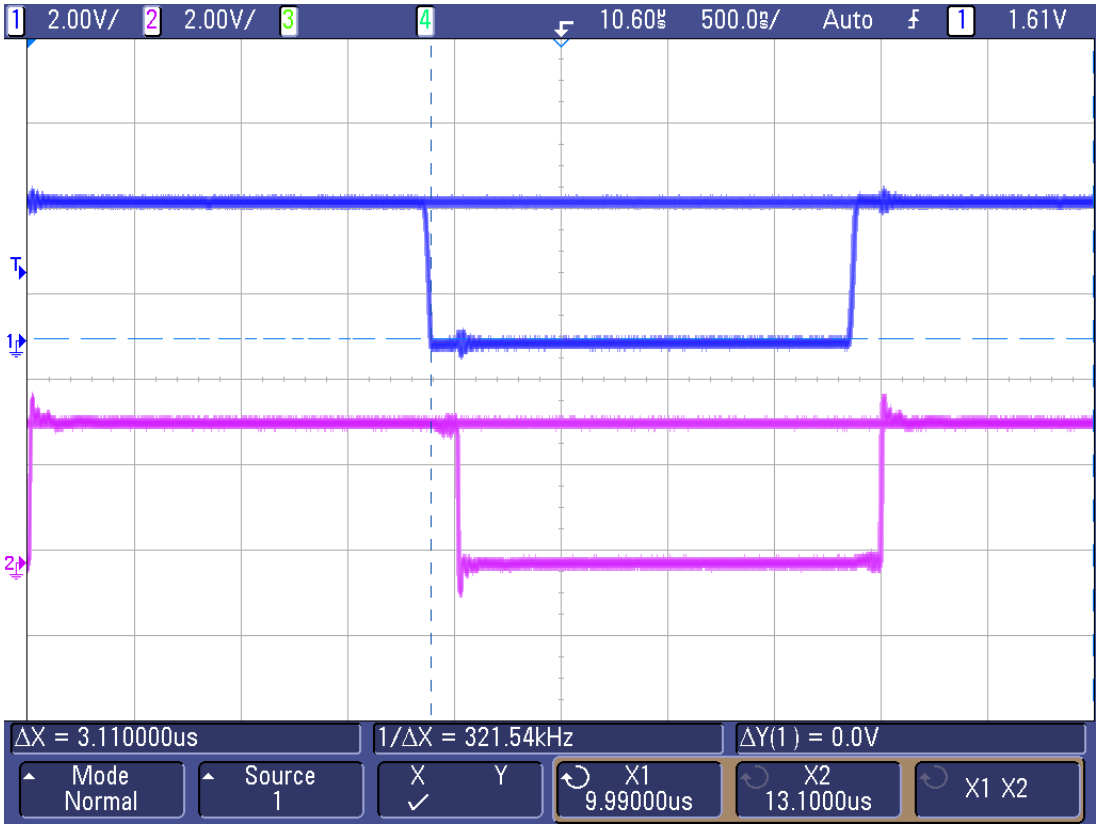


Figure 25: Test, Eight Sensors, 0 ft Distance, Delay Added Initial Reading

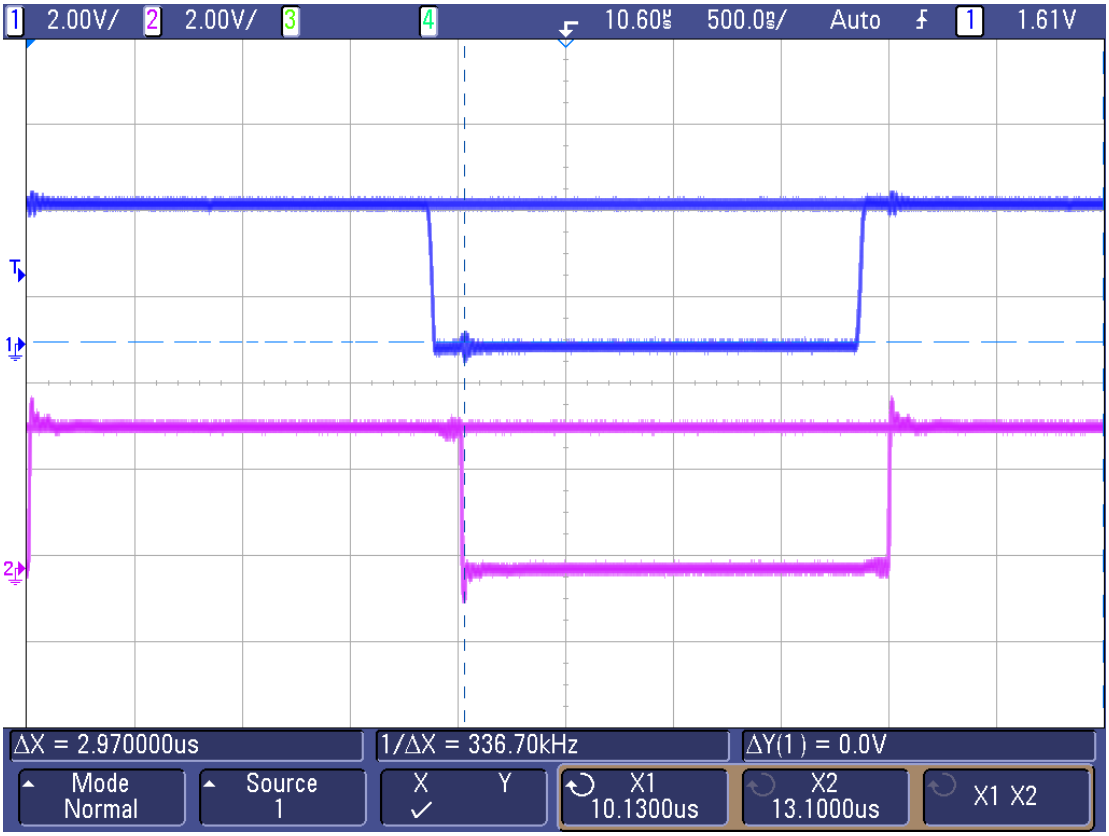


Figure 26: Test, Eight Sensors, 0 ft Distance, Delay Added Second Reading

This is showing that there is a delay caused by the eight buffers but it is only .14us which is not enough of a delay to cause issues with communication to the sensor nodes. When all of the signals required are sent through the buffer they will have the same delay, also negating any problem caused by a delay introduced by the buffers. The bottom signal is the signal is the SPI clock being sent from the controller node, and the top signal is the signal after it has passed through 8 buffers.

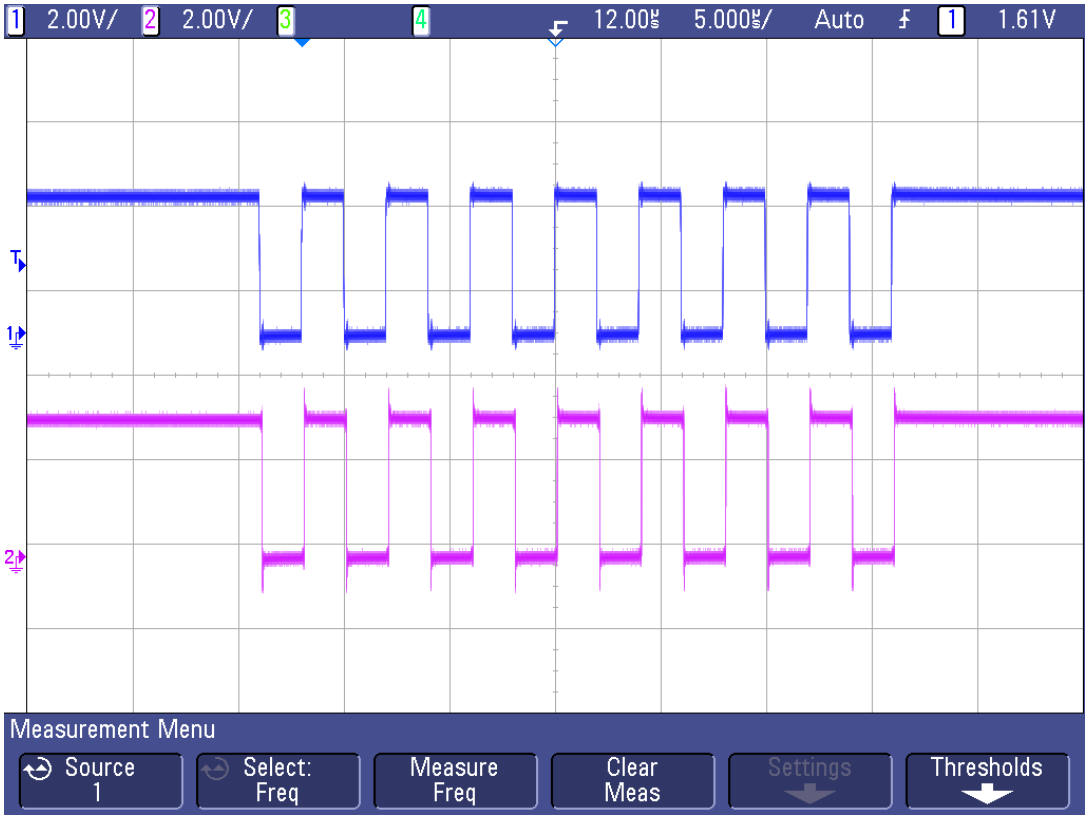


Figure 27: Test, Eight Sensors, 10 ft between each Node, Total 80 ft Distance

This is showing the effect on the signal after the signal has passed through 8 buffers and also through 80 feet of wire. There is no loss of signal with the added wire and the buffers. With the wires added it adds a delay of only 0.05us to the 0.14us of delay that was added with the buffers. The bottom signal is the signal is the SPI clock being sent from the controller node, and the top signal is the signal after it has passed through 8 buffers.

21.2.2.4 Parallel Communication over Distance (Sever / Controller Node)

Since all four controllers share the same TX/RX UART lines from the server, testing needed to be done to ensure the load of all four controllers being connected would not interfere with the communication even though only one would be transmitting at a time.

We also determined the longest distance our wires may could be was ten feet. We needed to be sure that with all controllers at a max distance combined with the load of all the controllers connected, the signal would not be lost.

Our test consisted of connecting 4 controllers to the TX/RX UART wires at a length of ten feet each.

We found that the system still worked with the distance and load attached. We connected several points to the scope to see how the signal was affected from the added load and distances. There was a noticeable difference, however the signal was still clear and acceptable enough to not need added buffers to improve it.

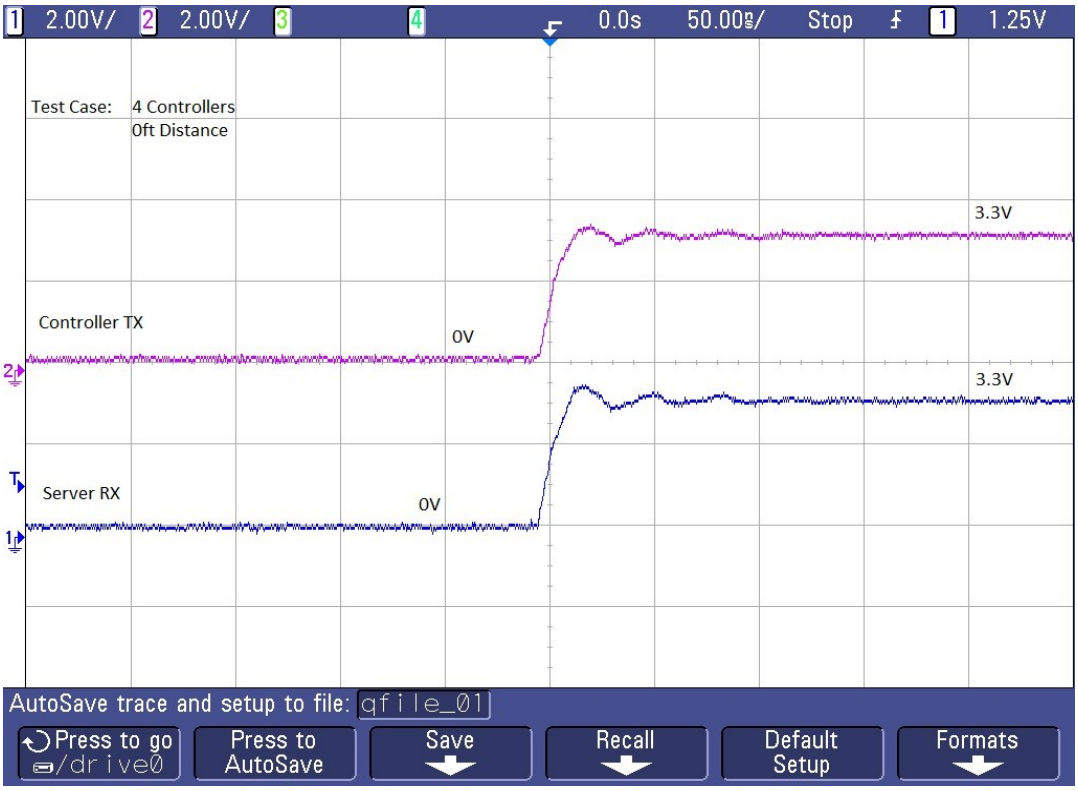


Figure 28: Benchmark, Four Controllers, 0 ft Distance

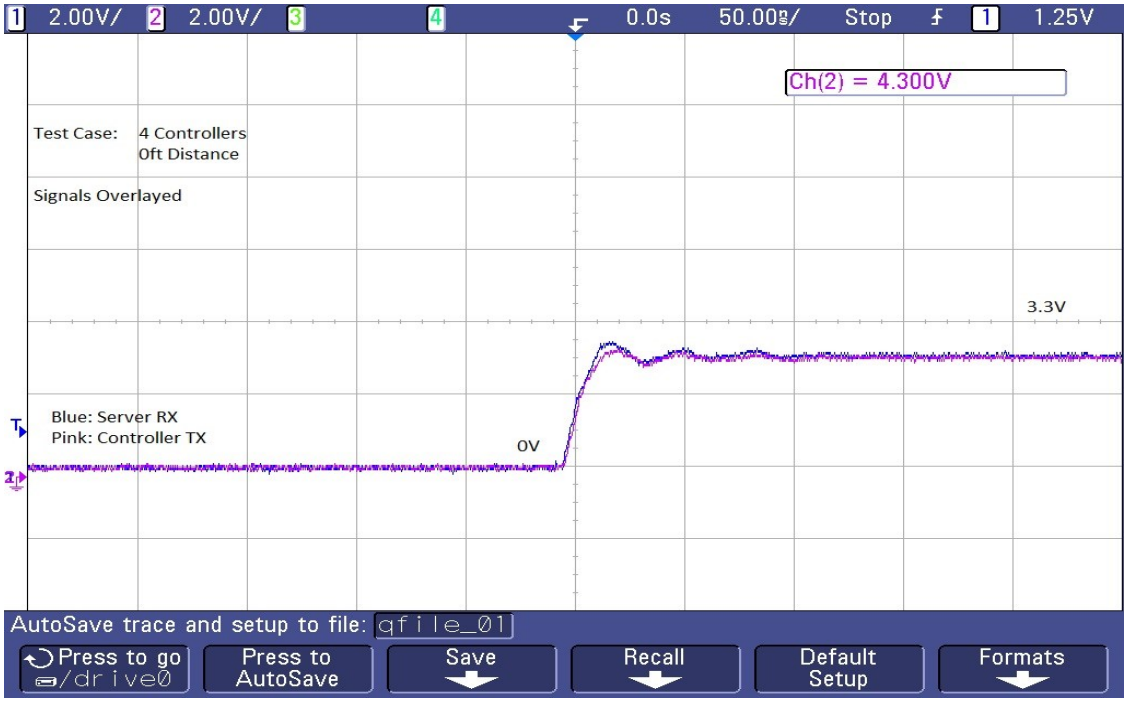


Figure 29: Benchmark, Four Controllers, 0 ft Distance, Overlapped

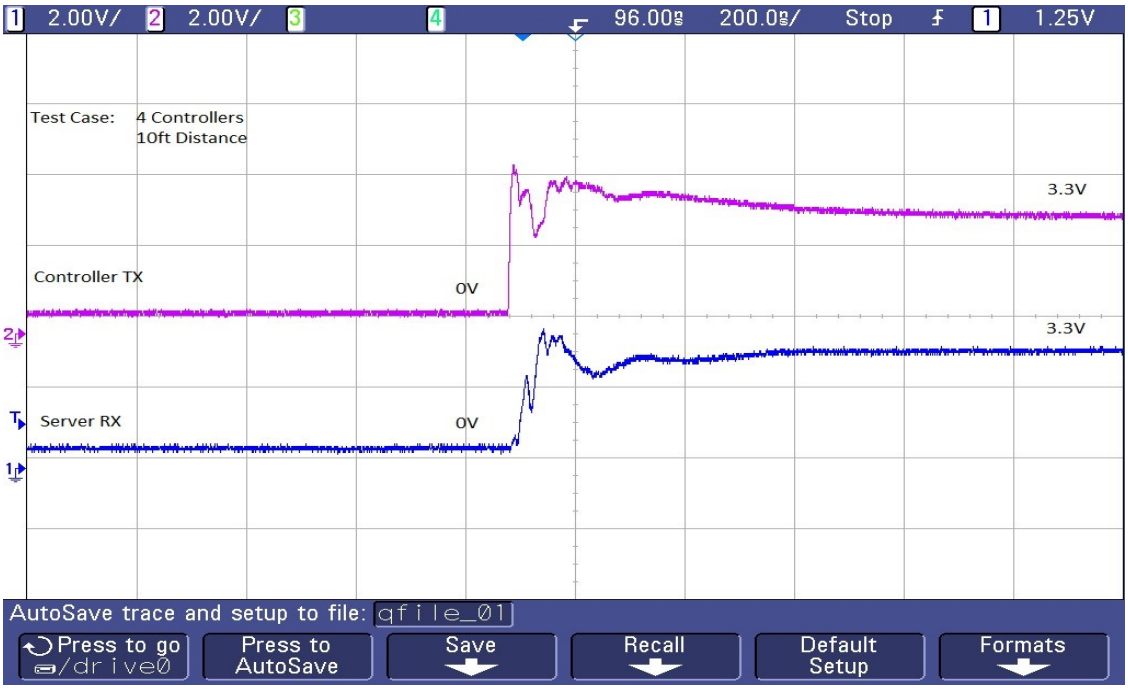


Figure 30: Test 1, Four Controllers, 10 ft Distance

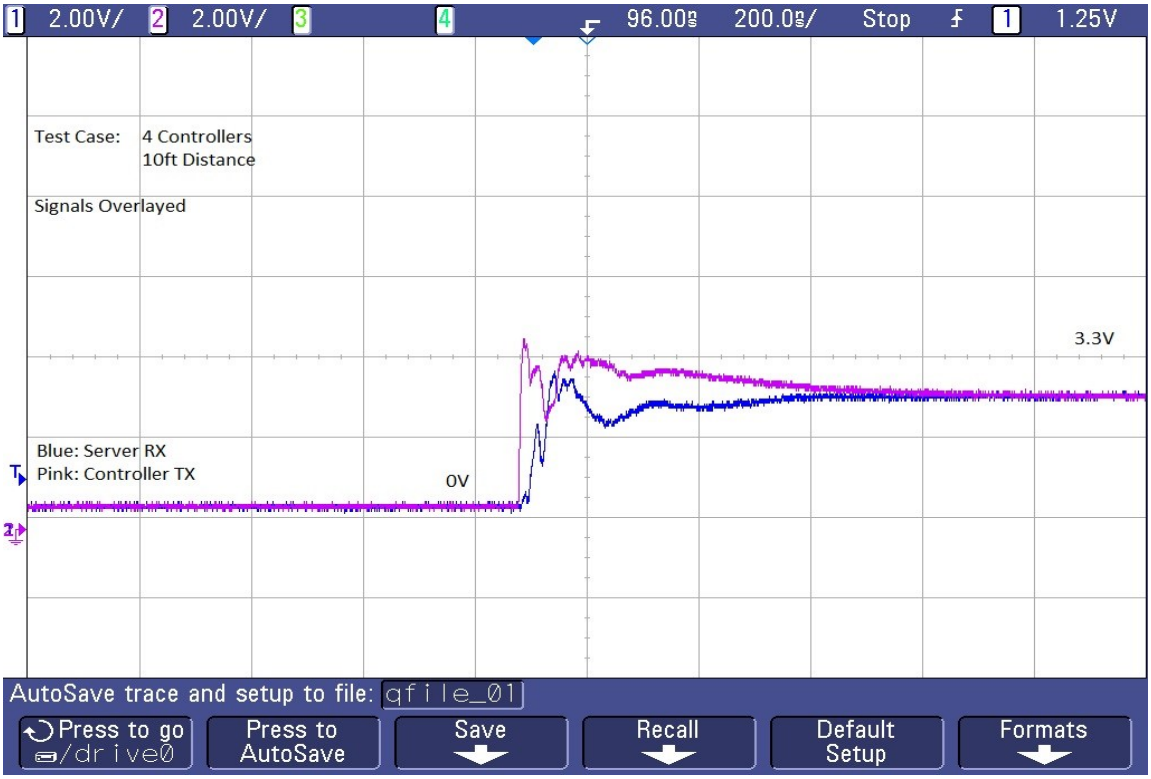


Figure 31: Test 1, Four Controllers, 10 ft Distance, Overlapped

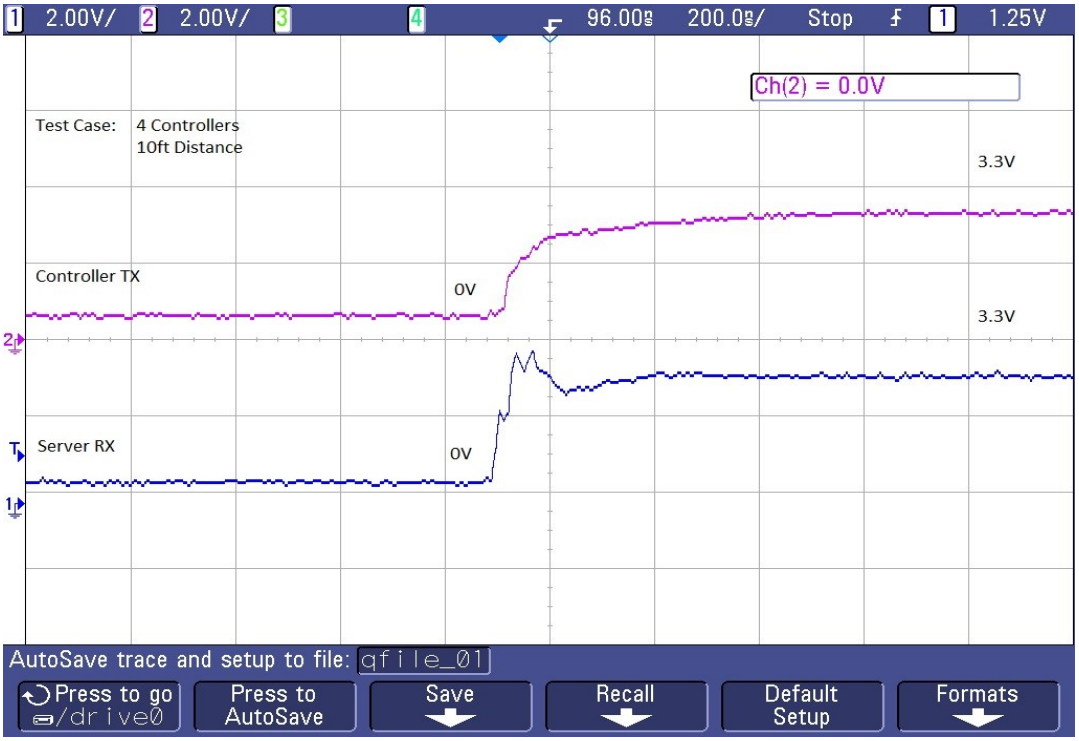


Figure 32: Test 2, Four Controllers, 10 ft Distance

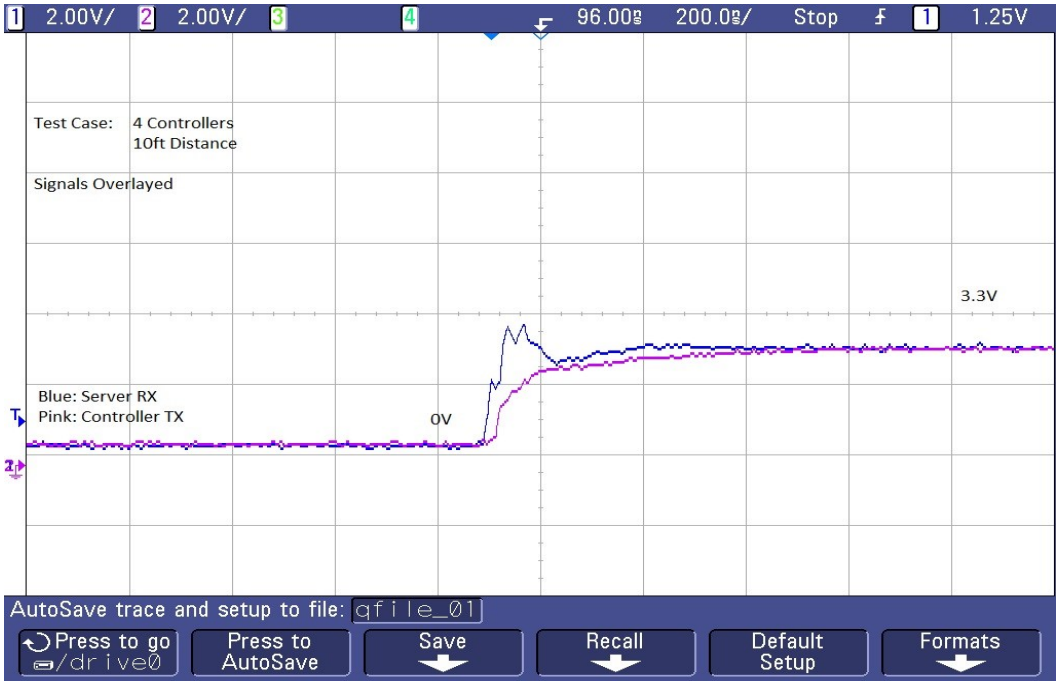


Figure 33: Test 2, Four Controllers, 10 ft Distance, Overlapped

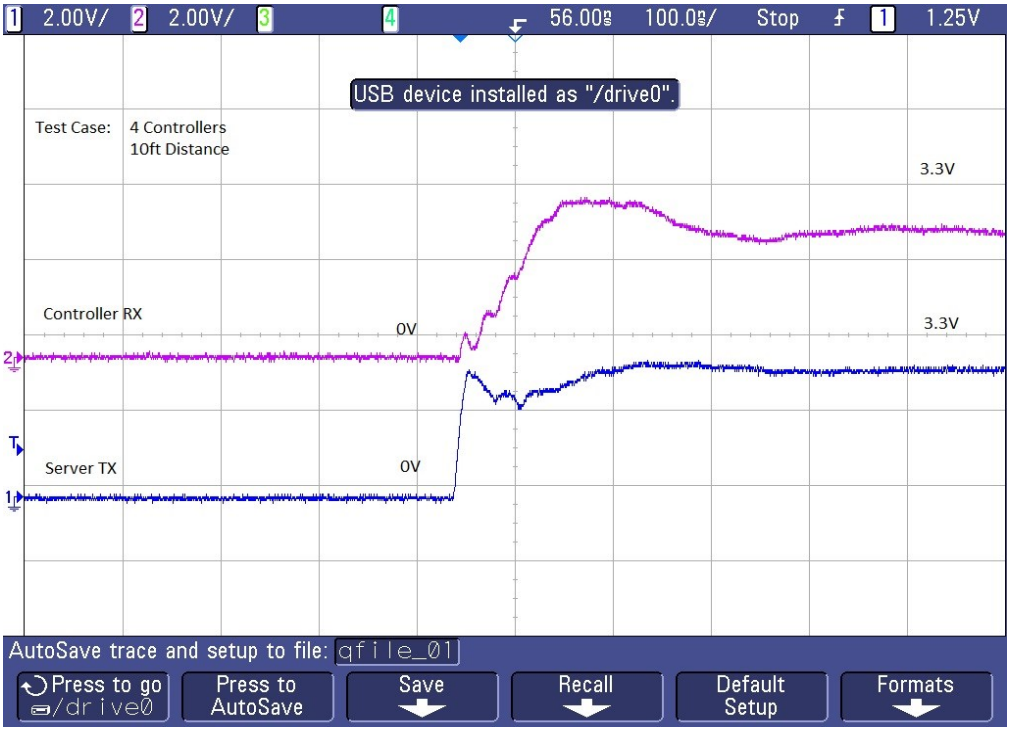


Figure 34: Test 3, Four Controllers, 10 ft Distance

21.2.3 On site Testing

Laundry Now on site Test Results			
Test	Other Test Notes	Test Conclusions/Results	Graph
Sensor Placement	Physical placement of the sensor on the machine resulting in the most transfer of movement.		See Figure
- Back Top		Back top of the machine provided the best transfer of movement, was the least in the way of normal operation of the machine.	35
- Side Center		Side center provided adequate transfer of movement, but was not as high as the back top. The sensor was also in the way of neighboring machines and would be difficult to install under certain circumstances.	35
- Top Center		Top center provided the least transfer of movement of the areas tested. It also was in the way of normal operation of the machine.	35
Machine Placement	Testing of different types of machines found in laundry rooms.		See Figure
- Stacked Dryers		Laundry Now is unable to determine which machine is running and which is not with stacked dryers and washers. In its current state, the algorithm does not compare movement profiles between sensors. By analyzing the graph it is clear to see which machine is running, but since comparison between sensors is currently not implemented the system does not support this.	42
- Washer/Dryer on Platform		Laundry Now is unable to determine which machine is running and which is not with a washer and dryer mounted on a raised platform. Though the transfer of movement was less than with stacked machines, there was still too much noise when the machine was not running, which caused false triggers. If the capability of comparing sensors was added to the controller, this type of machine could be supported.	

- Washer/Dryer on Rail		Laundry Now supports machines mounted on a single common rail attached to the floor. The noise is minimal and does not cause false triggers on neighboring machines	
- Washer/Dryer on Floor		Laundry Now supports machines next to each other on a solid floor. There was almost no transfer or noise between machines and no false triggers were caused.	
Time Delay (Sensor Reading)	Test to determine how much of a delay between two sensor readings will provide the largest intensity reading for analysis. The highest intensity will provide the most range of movement for analysis. Tested on a dryer, tests completed back to back with full loads of clothes.		See Figure
- 100 us		Average Intensity of 43.6468	
- 500 us		Average Intensity of 52.9636	
- 1 ms		Average Intensity of 50.6360	
- 10 ms		Average Intensity of 52.6116	
- 20 ms		Average Intensity of 42.4804	
Averaging Points (Sensor Reading) - Washer	The number of data points collected from the sensor to be averaged for use in the algorithm. Averaging is required to remove large spikes and dips in the data that will trigger the thresholds used to determine if a machine is running or not. Too little smoothing of the graph results in too many undesired spikes and dips that result in false triggers. Too smooth of a graph and the algorithm cannot detect key changes in the machine's movement.		See Figure
- No Averaging Data Points		There are too many spikes and dips that will falsely trigger the thresholds in the algorithm.	40
- 10 Data Points		Better than with no averaging, but there are still too many large dips and spikes from the movement that will cause false triggers	40
- 20 Data Points		20 data points provided the most accurate representation for use in the algorithm. This is the averaging Laundry Now uses for the washing machine algorithm	40
- 50 Data Points		50 data points smoothed the graph too much and the algorithm was unable to detect key movements	41
- 100 Data Points		100 data points smoothed even more	41
- 200 Data Points		200 showed no sudden spikes or dips, and consumed too much memory on the microcontroller	41

Averaging Points (Sensor Reading) – Dryer	The number of data points collected from the sensor to be averaged for use in the algorithm. Averaging is required to remove large spikes and dips in the data that will trigger the thresholds used to determine if a machine is running or not. Too little smoothing of the graph results in too many undesired spikes and dips that result in false triggers. Too many averaged data points has the downside of using too much memory on the microcontroller that could be used for other operations.		See Figure
- No Averaging Data Points		Similar to the washing machine, the dryer with no averaging had too many spikes or dips. This resulted in constant random false triggering.	38
- 10 Data Points		10 data points still did not provide enough smoothing of the spikes and dips and resulted in false triggers within the algorithm	38
- 20 Data Points		20 data points was sufficient, but due to the constant nature of the movement of the dryer we chose to go with a higher averaging to account for differences in types of machines	38
- 50 Data Points		50 data points smoothed enough of the graph to provide consistent results across different types of machines	39
- 100 Data Points		100 data points did not cause any false triggers due to the nature of the movement in a dryer, but used much more memory.	39
- 200 Data Points		200 data points also did not cause false triggers, but used too much memory.	39

21.2.3.1 Best Mounting Location

We tested three different locations of the sensor on the dryer. Ideally, the best location would have a large range of values in order to best determine the status of the machine. The first location was on the back of the dryer (side that faces the wall), along the top center. This location provided the best movement data with the highest overall average of values. This location was also the least susceptible to outside interference such as the sensor being bumped.

The second location we tested was the side of the machine, in the center. This location provided the smallest range of values. The panel along the sides acted as a buffer from the internal movement. This may have been intentional from the manufacturer to reduce

vibrations and noise resulting from the movement within the machine. This location also could possibly interfere with machines on either side.

The third location tested was the top of the machine in the center. The range of values in this location was better than the side, but not as high as the movement range on the back of the machine. This location was also the least practical, and most susceptible to outside interference.

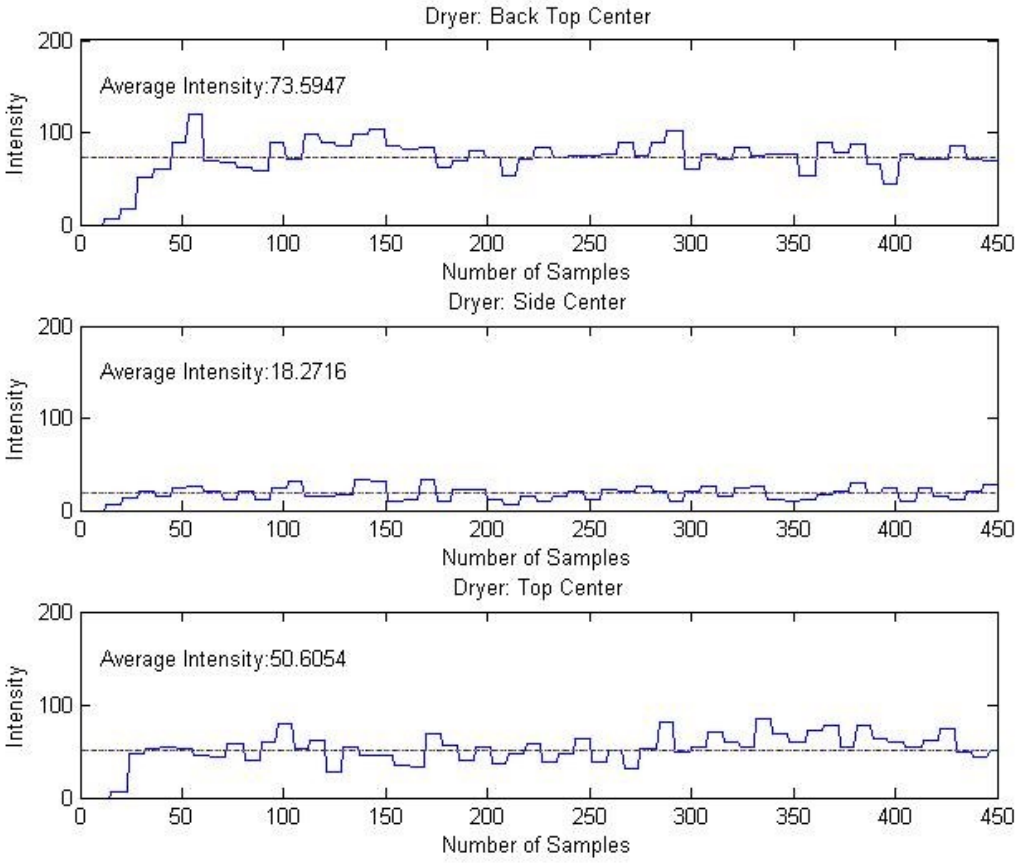


Figure 35: Sensor Mounting Locations

In the Process of determining the best mounting location for the sensor, we also found that the best measurements that could be achieved occurred with the sensor attached to the machine and not on a breadboard. In order for our system to work properly and reliably we will have to mount the sensor directly to the machine. In order to do this the

best means that we found was by attaching a magnet to the sensor and then having the magnet be the connection between the machine and the sensor.

It was also determined in this process that our system needs two separated system state detection programs, one that can be used for dryers and one that can be used for washers. With the current system state detection our product will detect the correct operating state of the system for driers but has problems with washing machines. This problem is caused by the wash cycle, which has larger portions of the run time with the machine not moving.

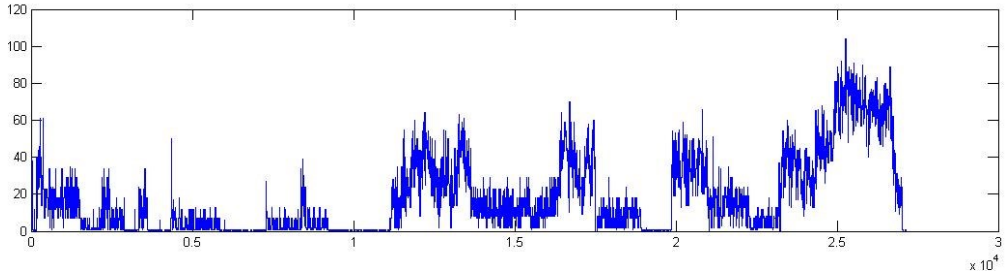


Figure 36: Full Wash Cycle

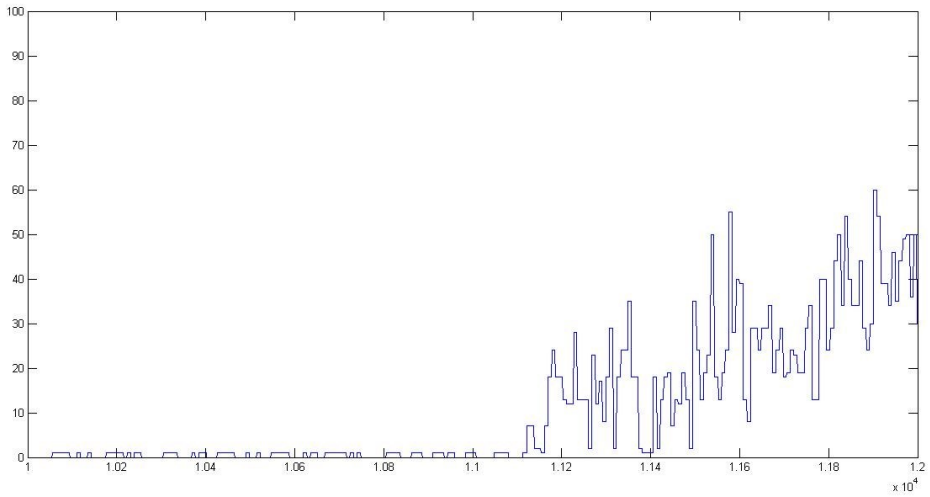


Figure 37: Zoomed-In Look at the Wash Cycle

21.2.3.2 WI-FI Distance

We tested the distance the WF32 could be from the router in two different scenarios. The first test was indoors and the approximate distance that was achieved was 50ft.

This test was through an interior wall and one floor below the router. The second test that we conducted was outside with no obstructions. The router was placed just inside a window, and the WF32 was able to be approximately 180 feet away.

21.2.3.3 Optimal Average

Determine the best average for determination of system state for washer and dryers. For averaging we will test 10 points, 20 points, 50 points, 100 points and finally 200 points.

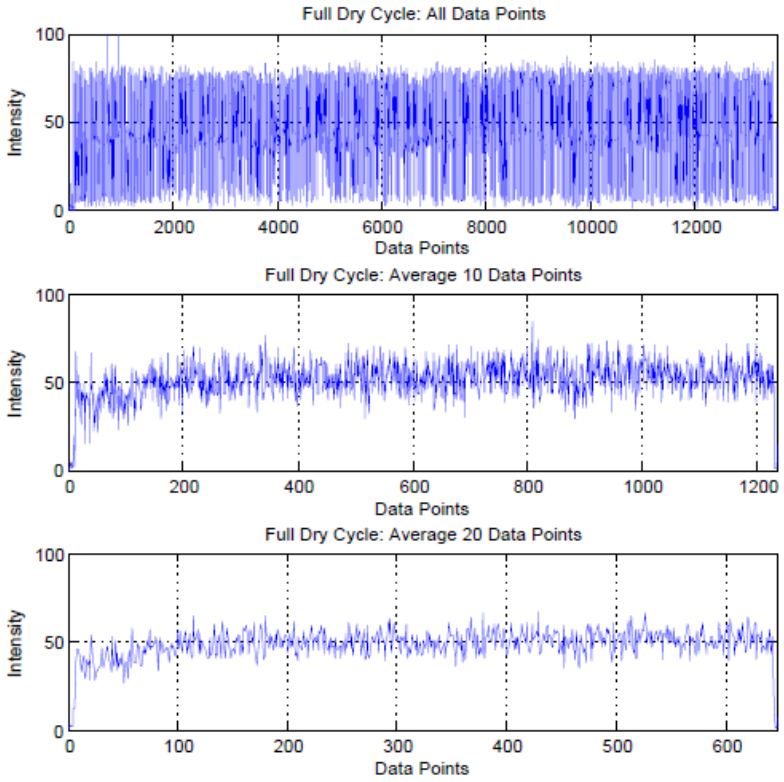


Figure 38: Dryer Averages 0, 10, 20

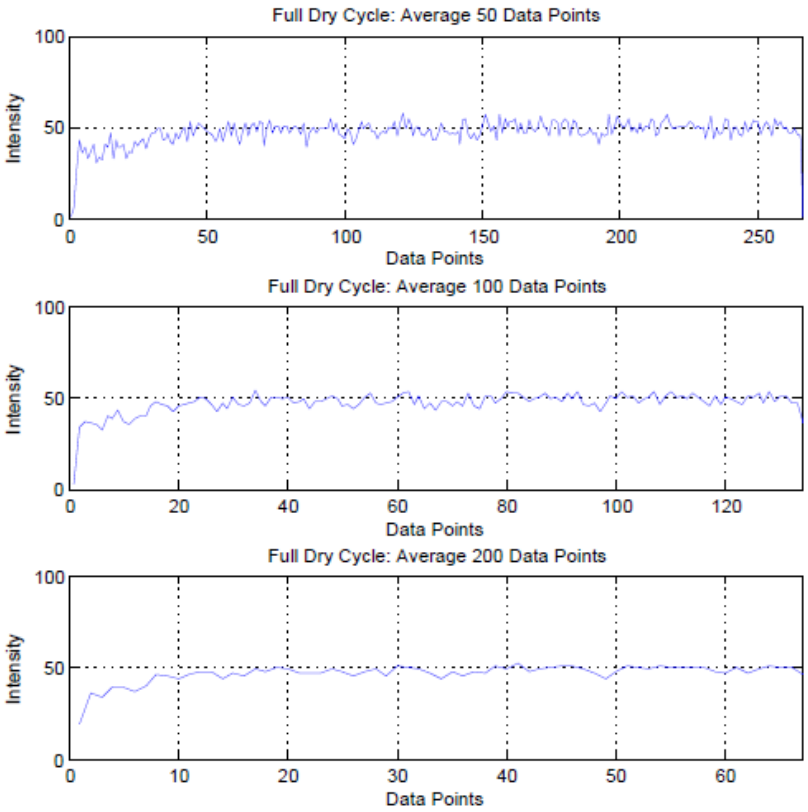


Figure 39: Dryer Averages 50, 100, 200

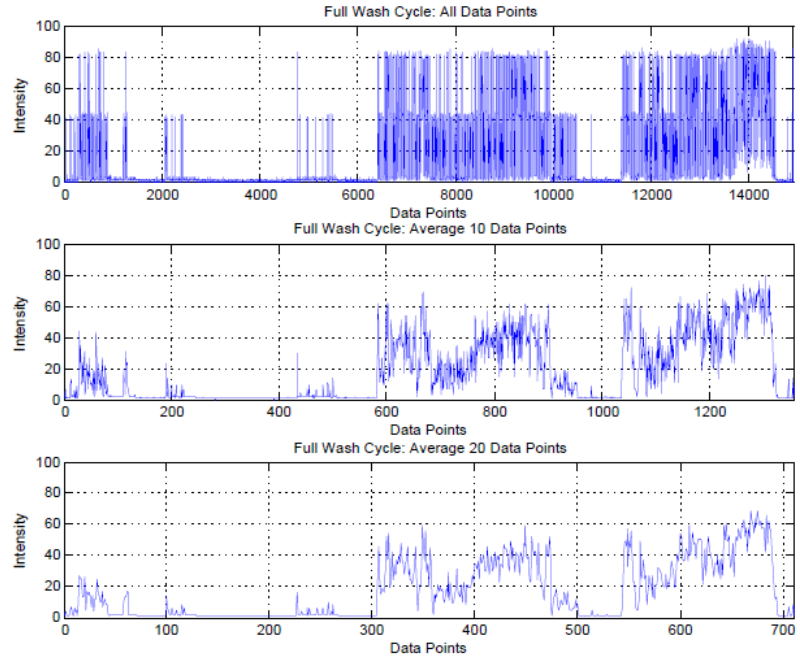


Figure 40: Washing Machine Averages 0, 10, 20

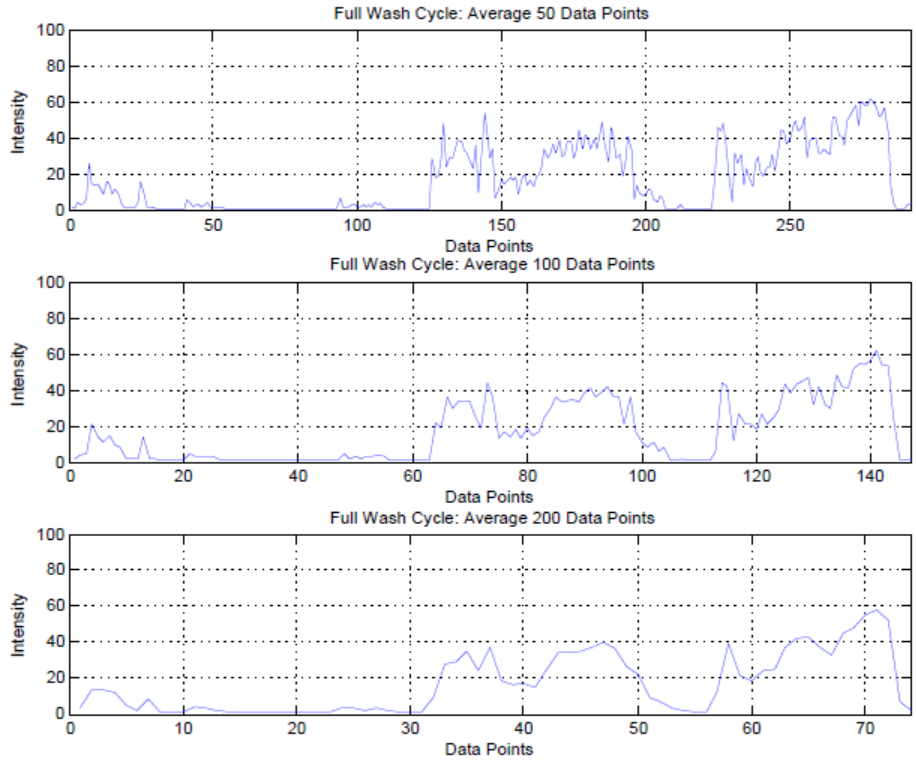


Figure 41: Washing Machine Averages 50, 100, 200

With this information we determined that the most optimal point average for dryers is a 40 point average and for washers we determined that the most optimal average is a 20 point average. With averaging not only is the operating frequency of the machines smoothed out, but false triggers also become less likely. For instance, when a sudden spike occurs during a sampling period the averaging will smooth out the sudden spike. A sudden spike could occur when someone kicks the machine or the machine is bumped into. This does not remove all false triggers, but it will reduce false triggers.

21.2.3.4 Machine Configuration

There are several different machine configurations that might be set up in different laundry rooms, from machines standing alone, machines touching each other, machines connected to each other (stacked or side by side), and finally stacked and side by side.

These tests were to determine which configurations allowed Laundry Now to work most effectively.

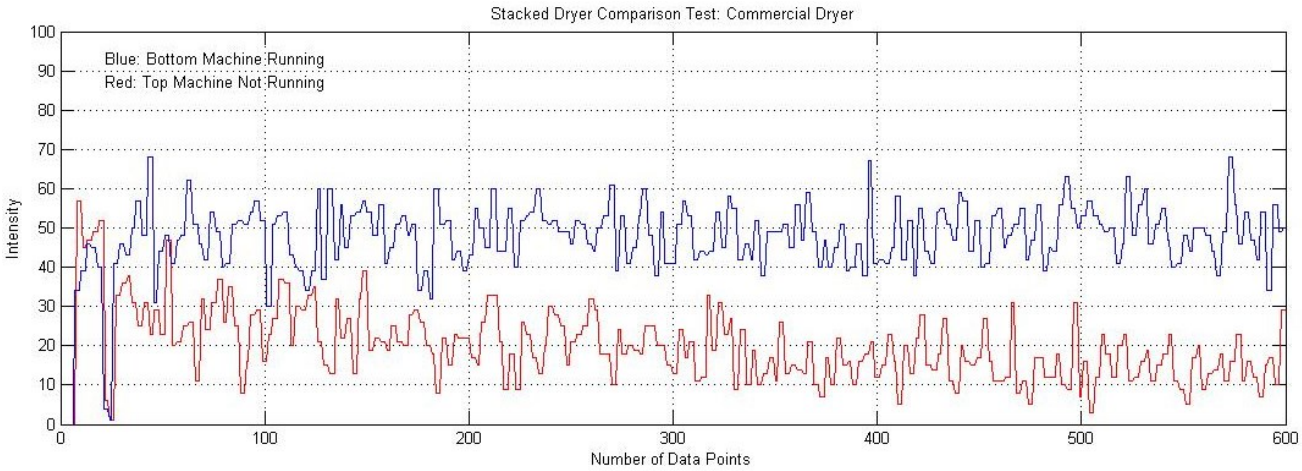
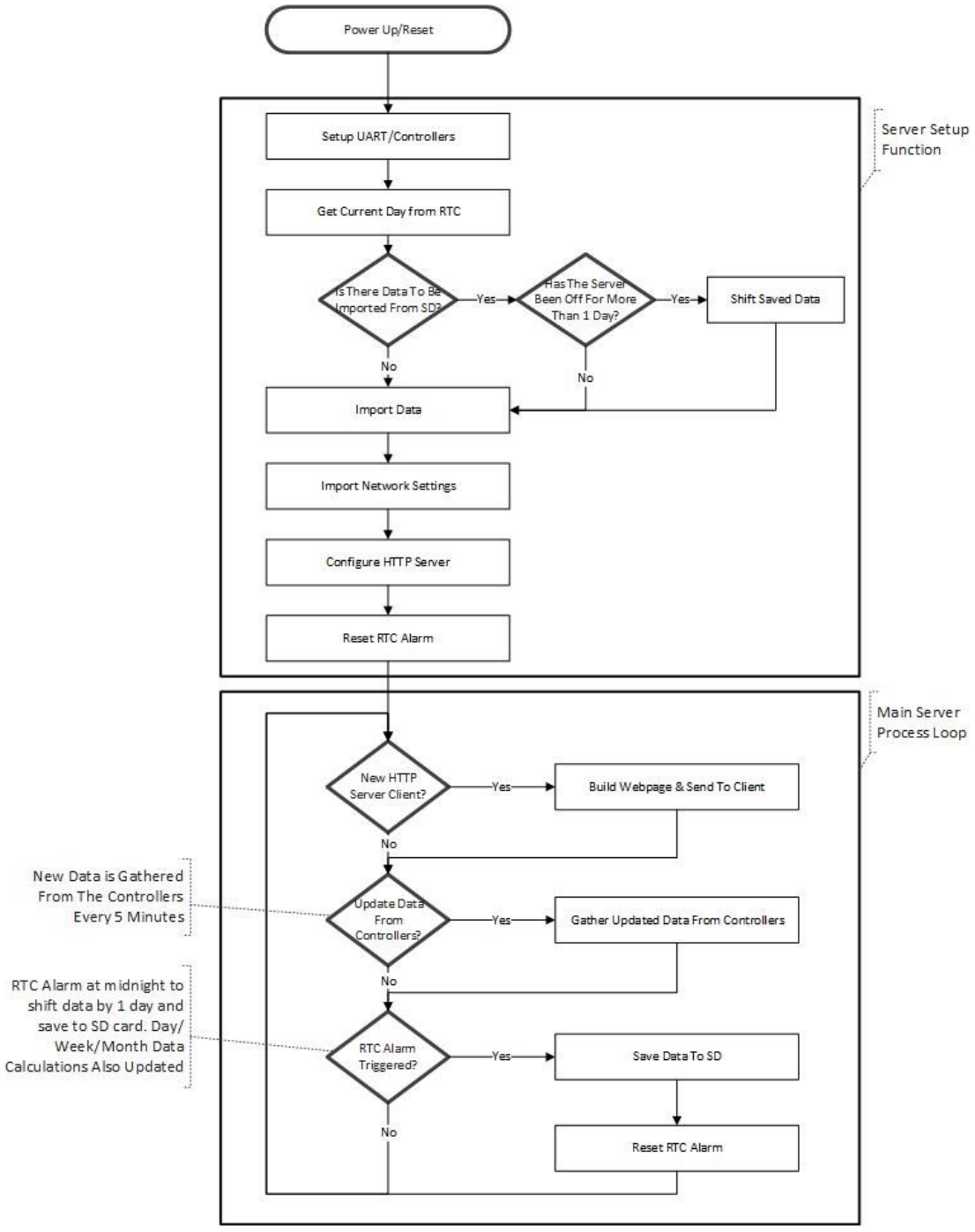


Figure 42: Test Results for Stacked Machines

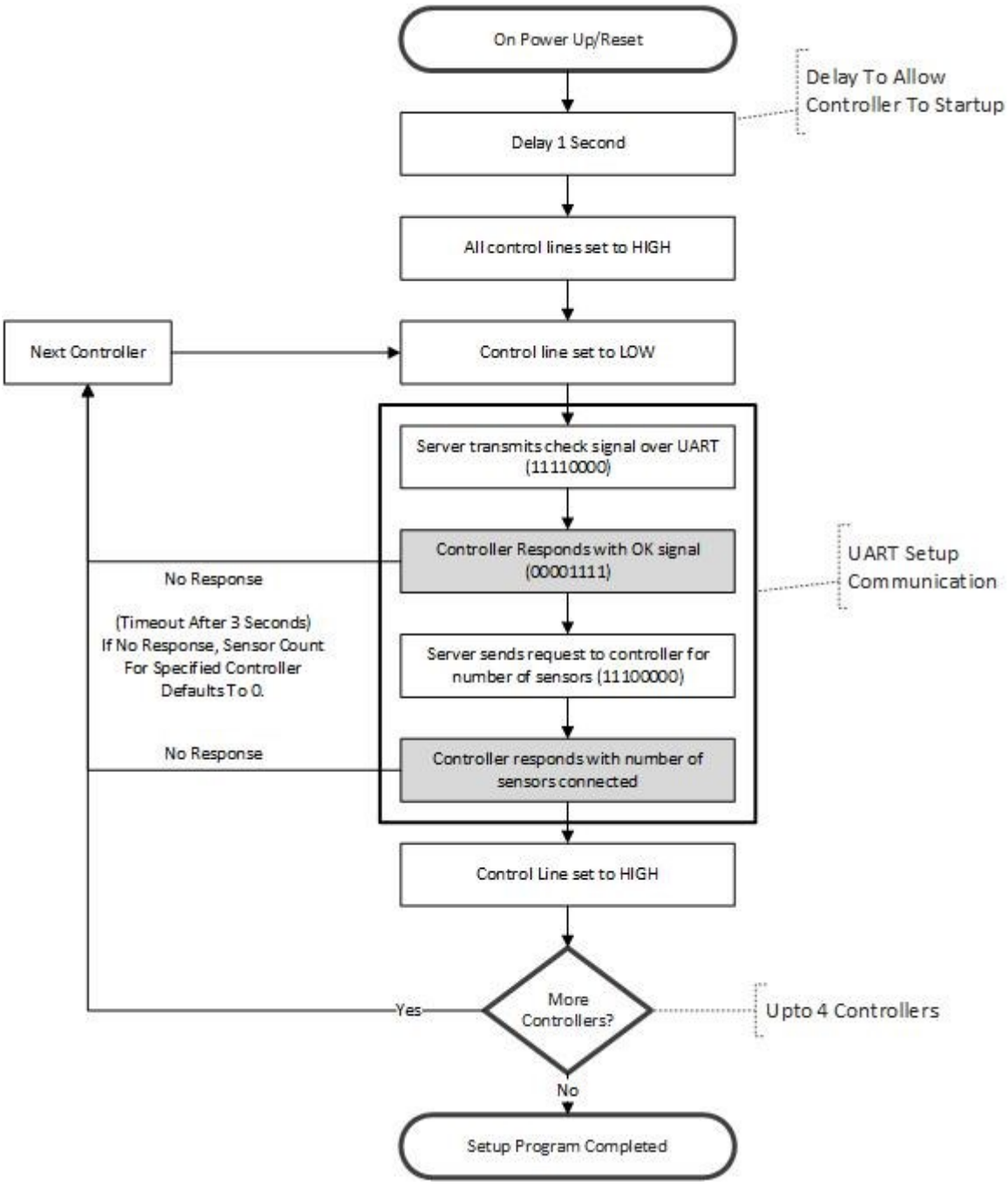
Currently Laundry Now cannot accurately determine which machine is currently in the operational state and which is not in an operational state. This is due to the noise generated by the machine in operation giving a false trigger to the machine not in operation.

21.3 FLOW CHARTS



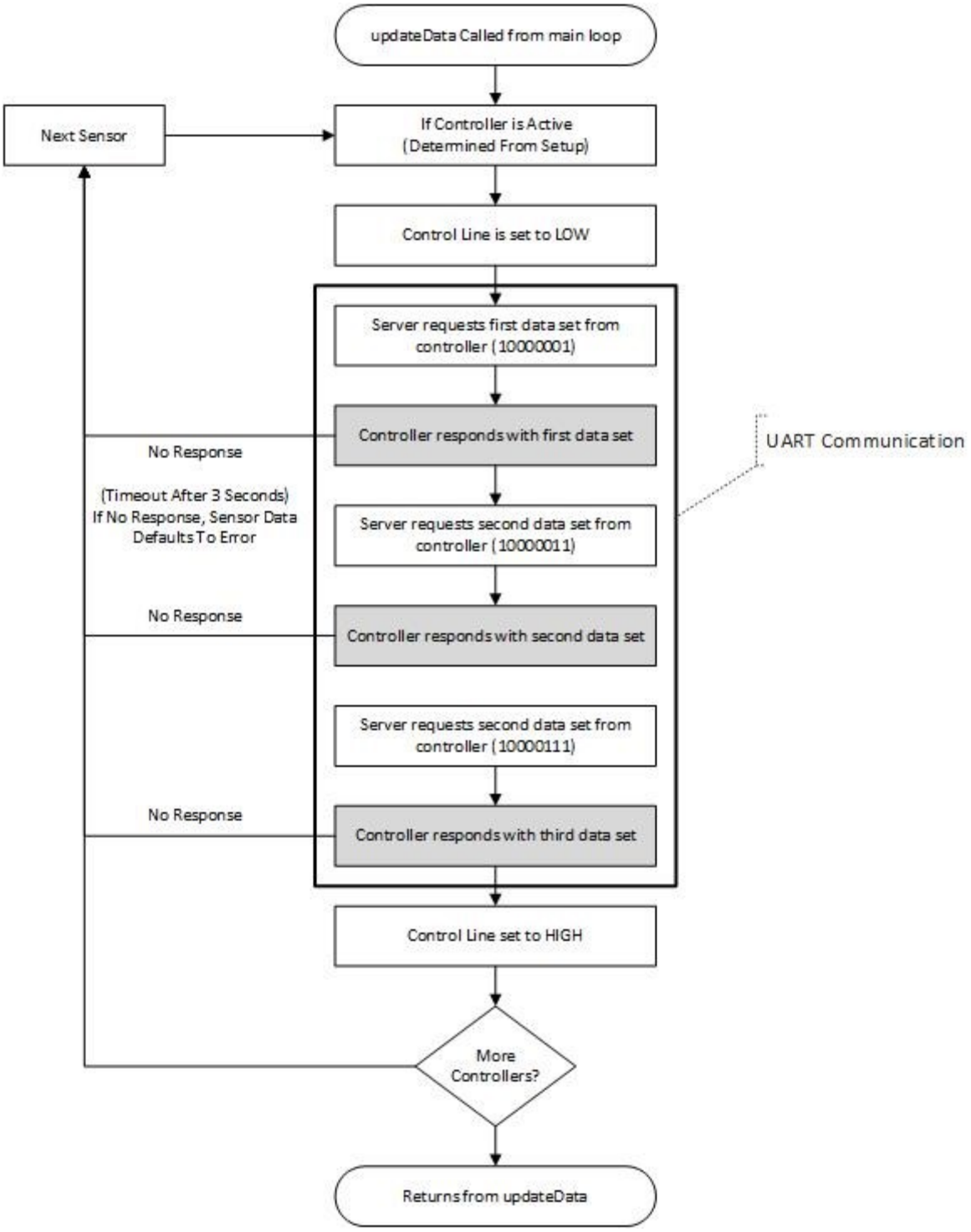
Flow Chart 1: Higher Level Server System Block Diagram

This flow chart depicts how the server operates, and the steps that it goes through throughout its operation. This depicts the initial communication between the controller node and the server in order to create the self-scaling website. It also goes through the process that the server takes after the initial setup.



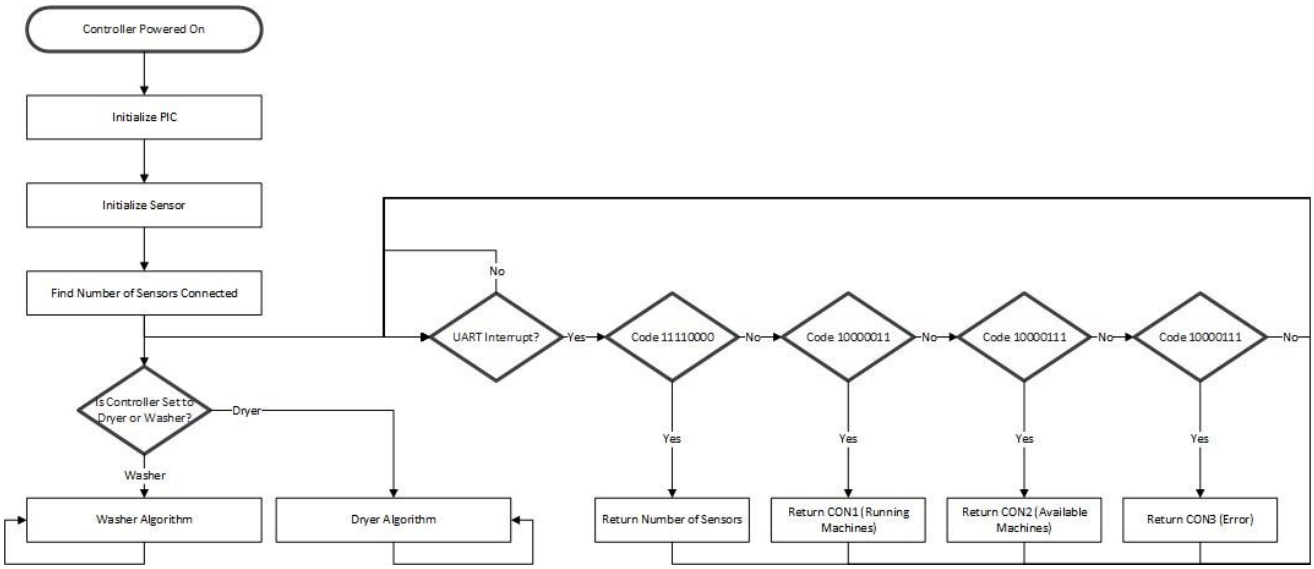
Flow Chart 2: UART Initial Request

This Flow chart depicts the initial communication between the server and the controller node. It depicts in more detail the handshake that occurs between the controller node and the sever node during this process.



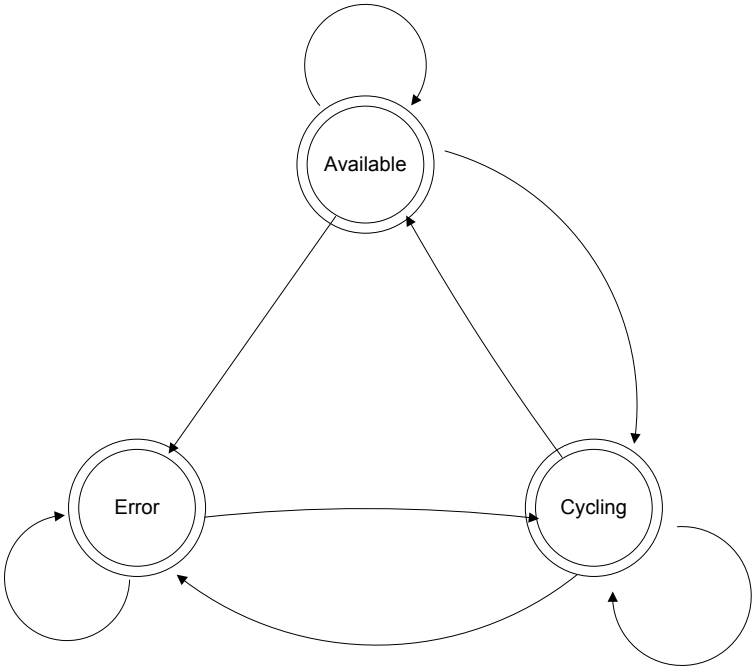
Flow Chart 3: UART Data Request

This flow chart shows in more detail the handshake that occurs every time the controller node and the server communicate with each other. This is how the data for which machine is in which state is transmitted.



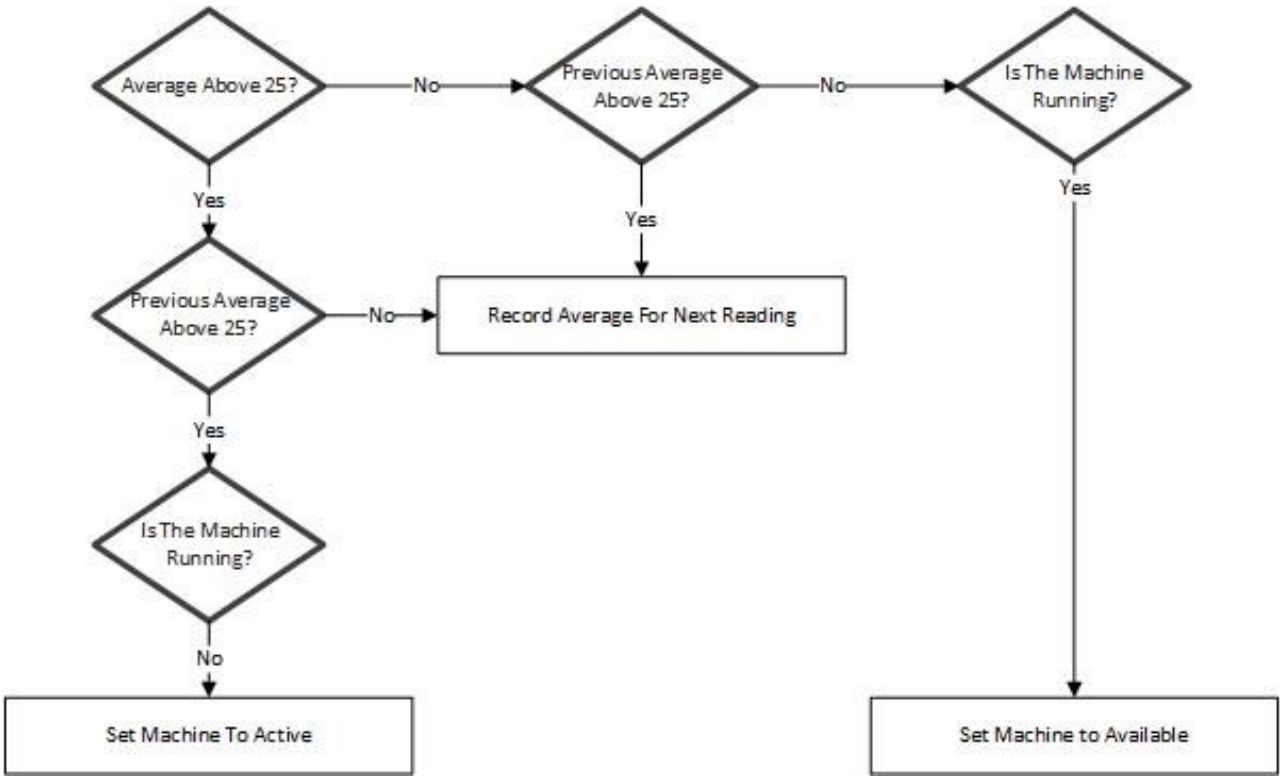
Flow Chart 4: Higher Level Controller Node Block Diagram

This is the higher level flow chart for the controller node. It depicts how the controller initial setup, followed by the controller node determines which algorithm to use either dryer or washing machine algorithm. It also depicts interrupts process that is used for the controller node to communicate with the server.



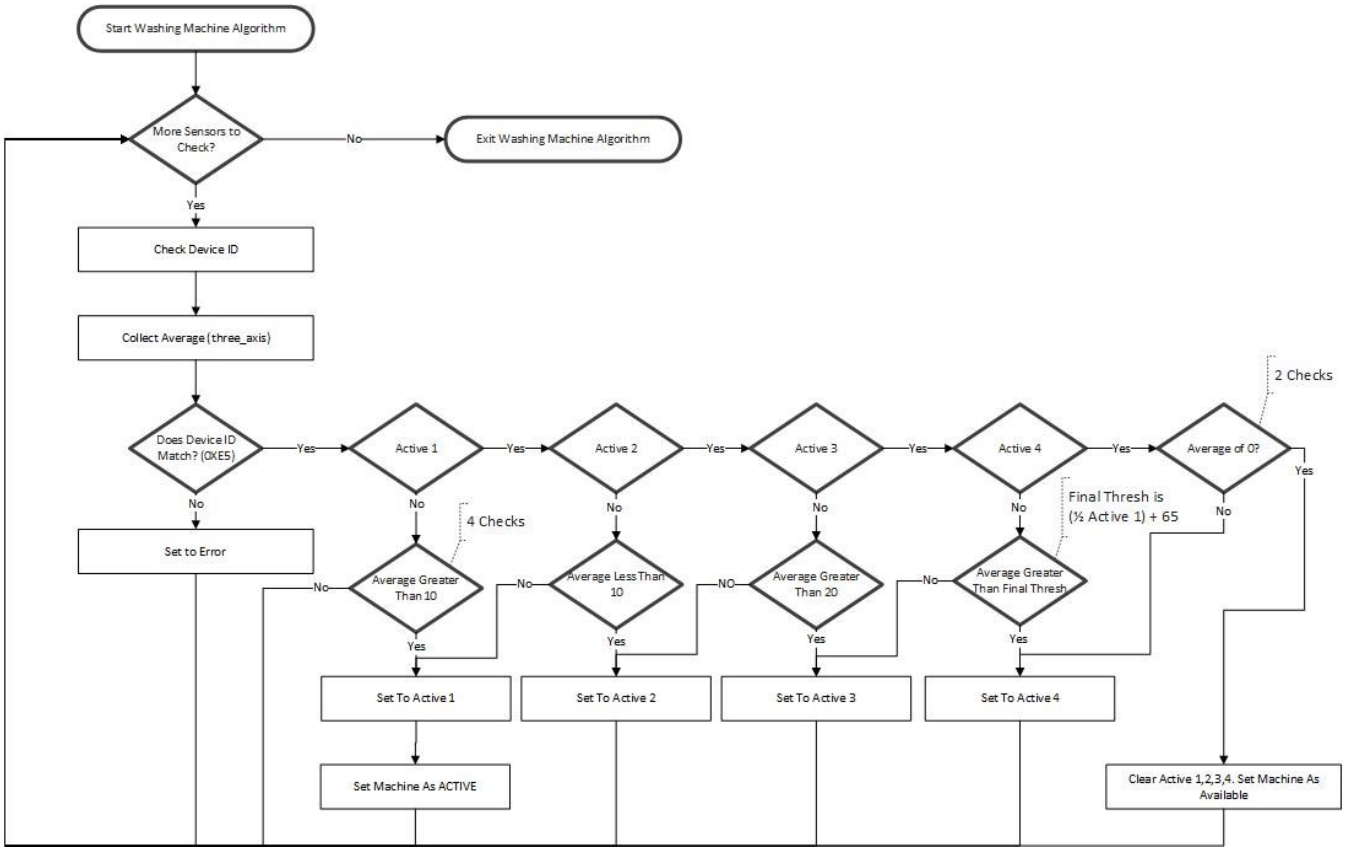
Flow Chart 5: Three State Diagram

This three state diagram shows how the controller node can change the system state of the machine that the sensor node is attached to.



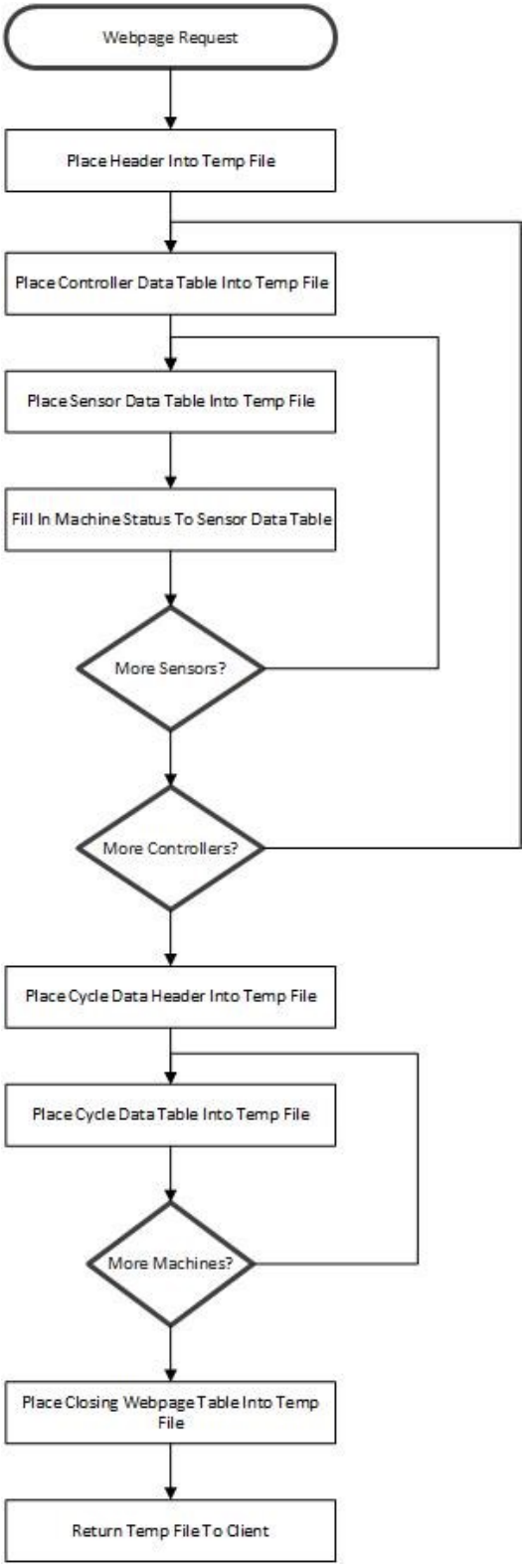
Flow Chart 6: Three State Dryer Algorithm

This flow chart shows how the dryer algorithm works for each machine. As long as it is above the threshold of 25, the machine will be in the operation state. If it is not above this threshold, then the machine will be set into available mode. If the controller node does not receive a device ID then the controller node will set the machine into the error mode.



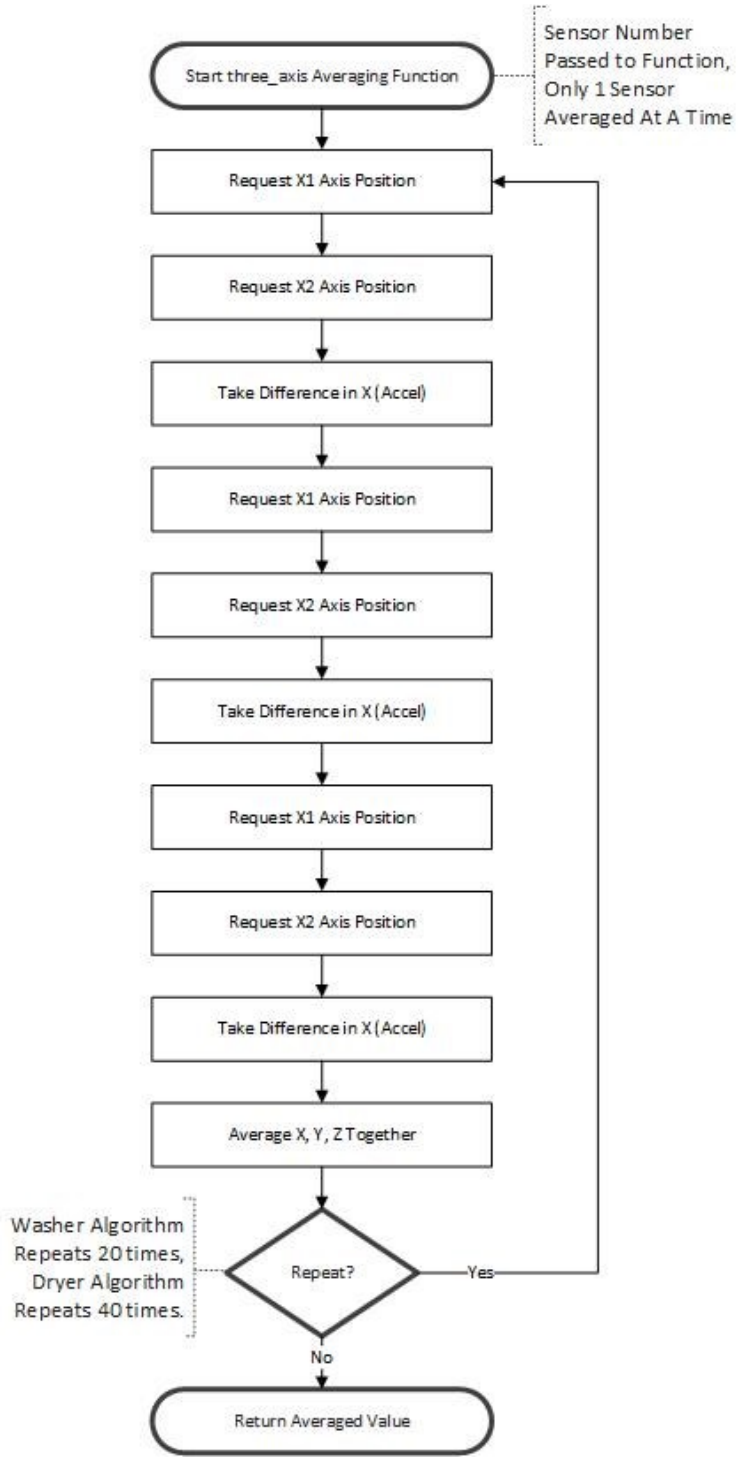
Flow Chart 7: Three State Washer Algorithm

This shows how the washing machine algorithm functions. The machine must be generating enough movement so that it is over 20, then the machine must be not running for a period of time; this is to capture the soak cycle. Finally, the system must detect a value above 65 in order for the machine to be set back at available. There is also error checking going on, first requesting the device ID before pulling the x, y and z axis.



Flow Chart 8: Webpage Build Process

This shows the website building processes that the server uses when first powered on.



Flow Chart 9: Three Axis Reading

This shows how the three axis average is conducted. The system takes an average over the x, y, and z axis and then takes an average several more times in order to generate the actual number which is then compared with the values in the dryer algorithm and the washing machine algorithm.

21.4 CIRCUIT DIAGRAMS

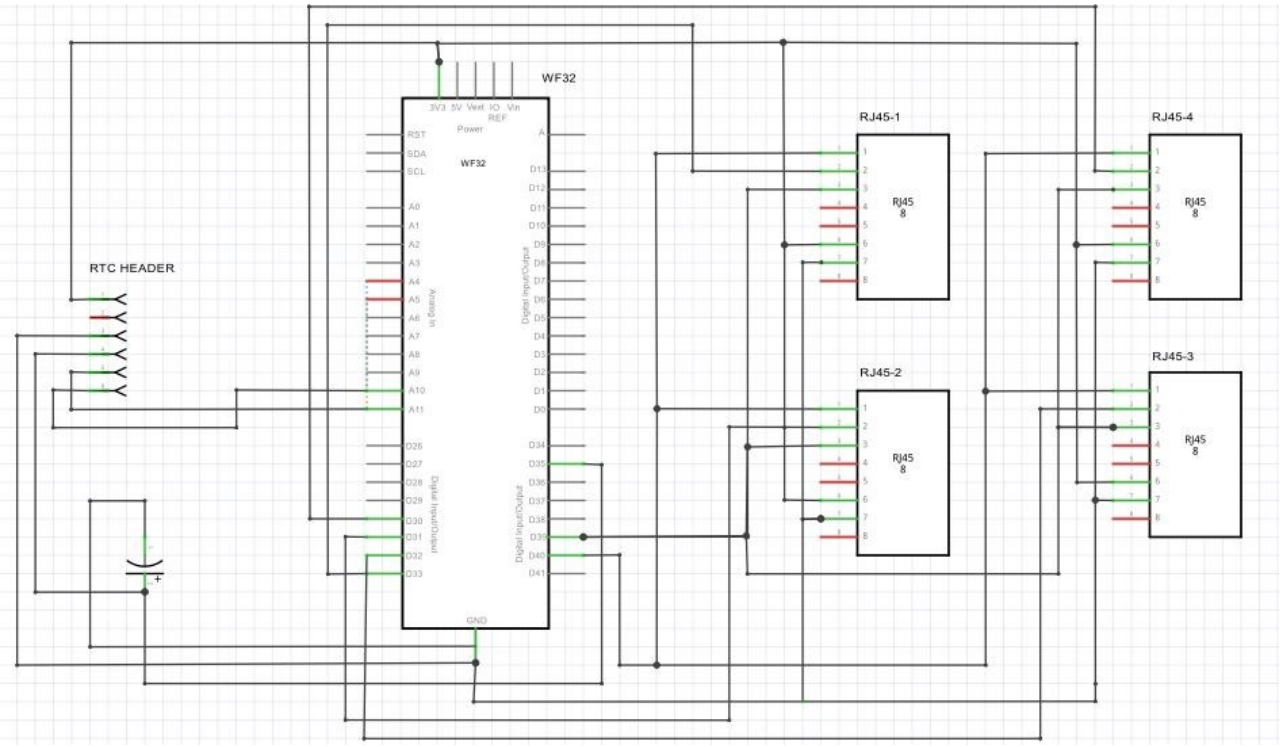


Figure 43: Server Schematic

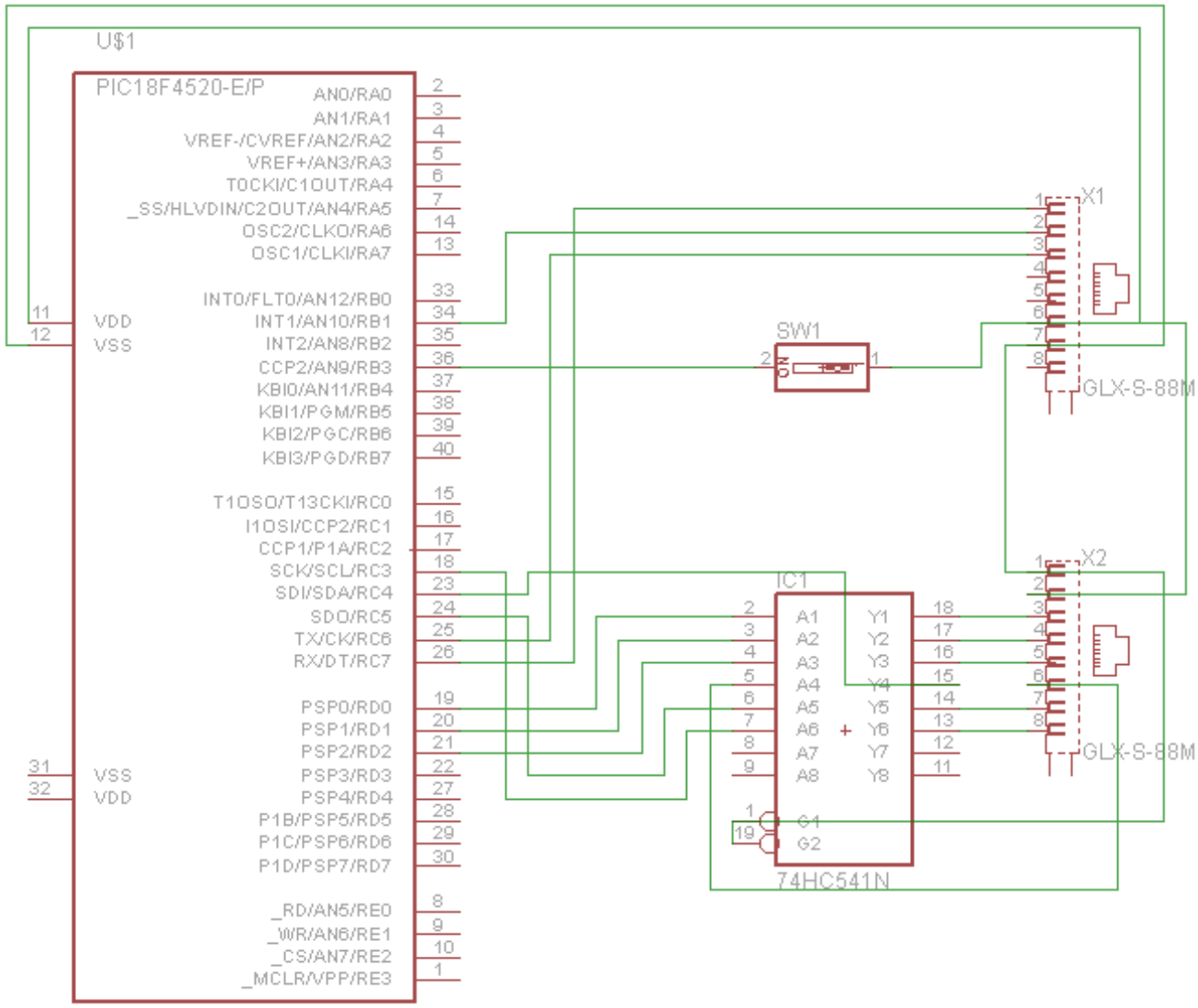


Figure 44: Controller Node Schematic

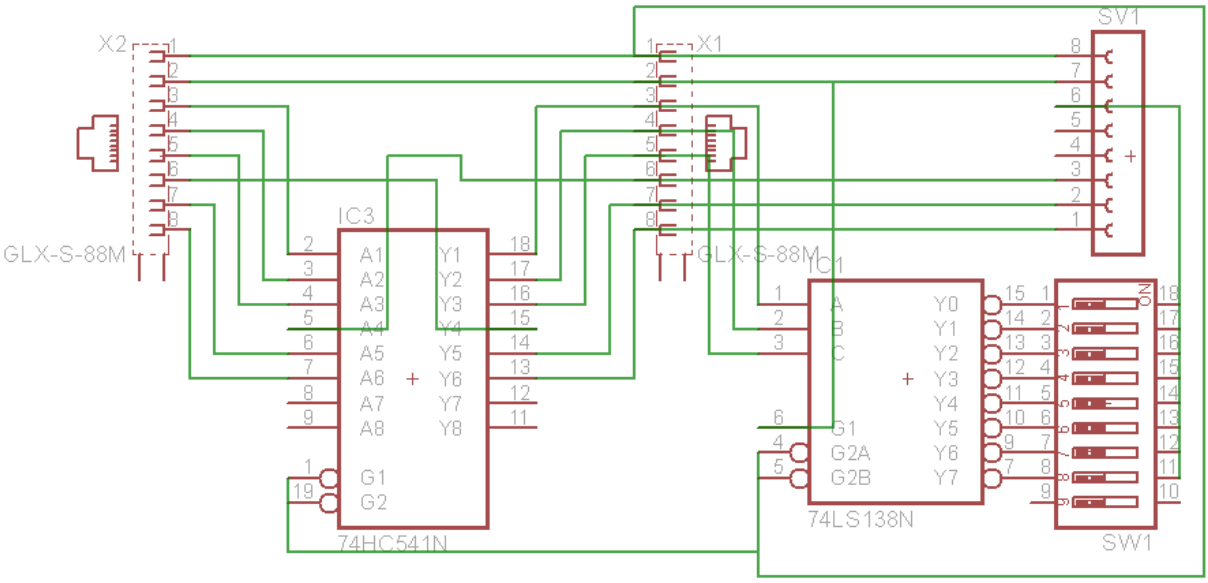


Figure 45: Sensor Node Schematic

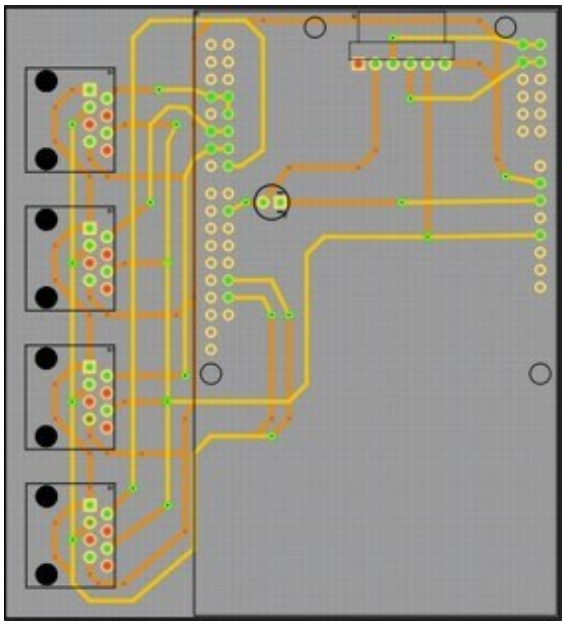


Figure 46: Server PCB Layout

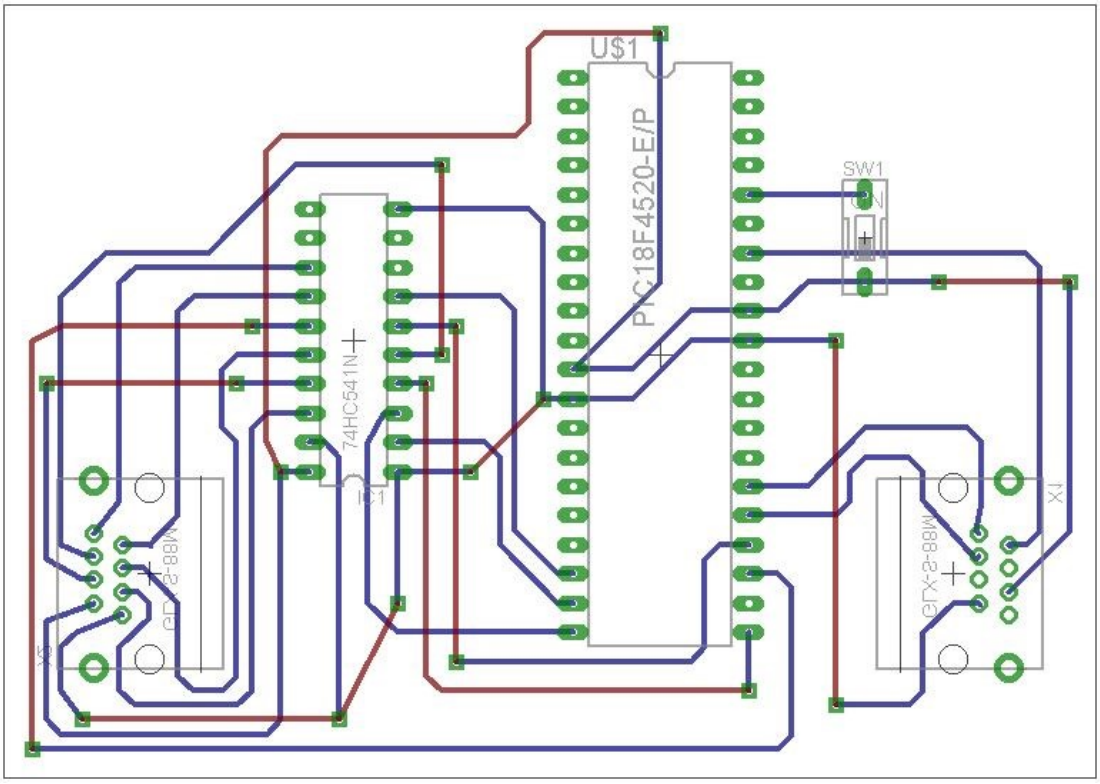


Figure 47: Controller Node PCB Layout

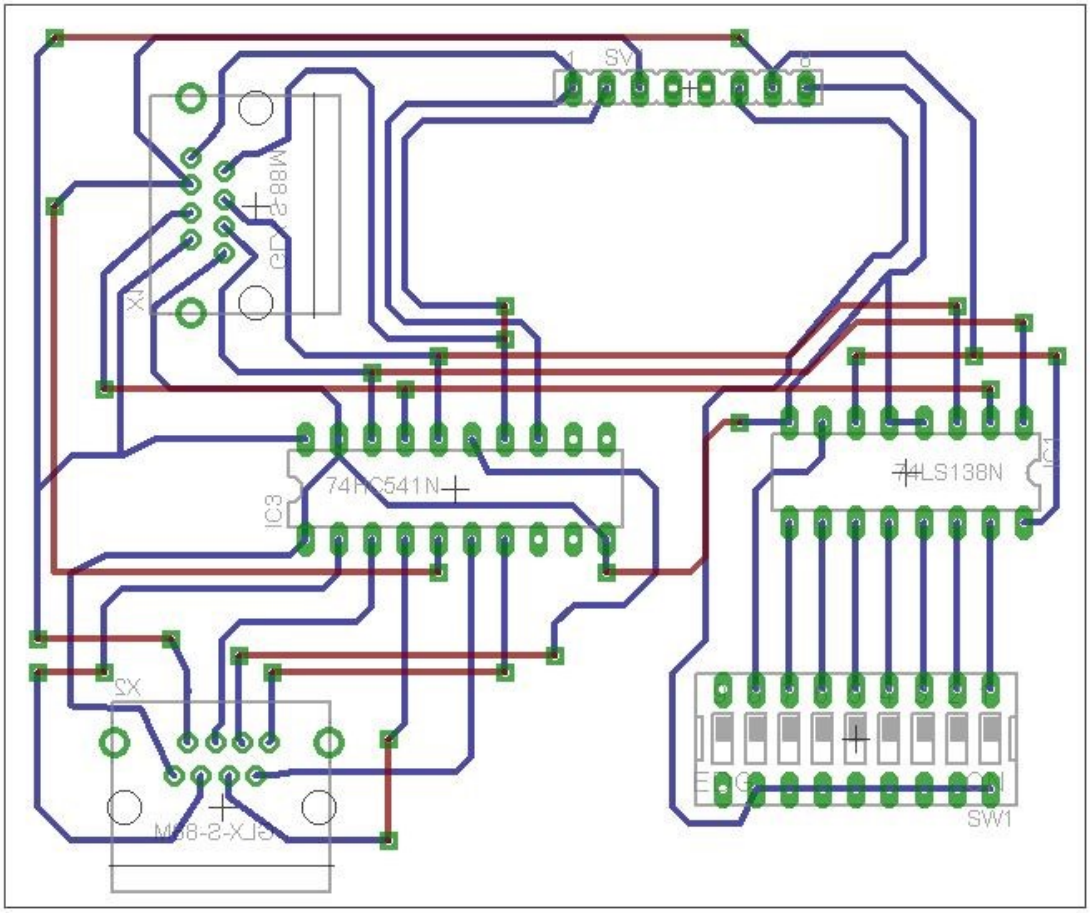


Figure 48: Sensor Node PCB Layout

21.5 BILL OF MATERIALS

Part	Cost per unit	Number	Total	With shipping
ADCL345, 3- axis Accelerometer	7.49	5	37.45	37.45
Chipkit- WF32, Sever	69	1	69	79.16
74HC7451, Buffer	2.34	10	23.4	30.44
74HC138, Demultiplexer	0.347	10	3.5	10.54
PIC18F45K20, Controller	2.9	3	8.7	15.74
Eithernet connectors	0.37	30	11.04	11.04
PCB Board, Sensor node	33	4	132	151.65
PCB Board, Controller node	33	2	66	85.65
PCB Board, Sever	33	2	66	85.65
Capacitor		1		Supplied by SSU Department of Engineering Science
Switch, 9 inputs		10		Supplied by SSU Department of Engineering Science
Switch, 1 input		4		Supplied by SSU Department of Engineering Science
Resistor, 150 Kohm		10		Supplied by SSU Department of Engineering Science
Micro SD card, 4 gig	6.46	1	6.46	6.46
Straight Through eithernet cable		6		Supplied by SSU Department of Engineering Science
			Total	513.78

21.6 In-depth Description of Technology and Components

21.6.1 ADXL345

Operates at 2.0 to 3.6 Volts

3-wire or 4-wire SPI communication

2 to 16 G in x, y, z axis

Capable of utilizing double tap, single tap interrupts in x, y, z axis

Capable of utilizing a FIFO (first in first out) up to 30 values store in x, y, z axis

21.6.1.1 Purpose

This chip is used as the accelerometer that is placed directly on the back of every machine in a laundry facility.

21.6.1.2 Implementation

This sensor is placed on the sensor node which is then mounted on the machines.

21.6.2 PIC18F45K20

21.6.2.1 Specs

Operates at 3.3 Volts

Capable of using SPI and UART communication

Up to 64 Kbytes of program memory, 3936 bytes of data memory

21.6.2.2 Purpose

This microcontroller determines the system state of the washer or dryer that it is connected to and then stores the system state. It also controls the communication between the sensor node and the controller node.

21.6.2.3 Implementation

The chip is implemented on the controller node and is the main MPU for the controller node themselves. It determines the system state of each machine and also controls the communication between sensors.

21.6.3 PIC32MX69F512L on a Diligent Breakout Board

21.6.3.1 Specs

Operates at 3.3 volt or 5 volt

Capable of SPI, I^2C, UART communication

Features onboard Wi-Fi module, micro USB capability

21.6.3.2 Purpose

This chip is used to create the website that Laundry Now utilizes to update usage data for the different machines and store former usage data. This board also has a Wi-Fi card which is used to access the internet via Wi-Fi connection. There is also an onboard micro USB which is used both to store former usage data and also store the router name and password. This chip also controls the communication between the controller node and the sever. This communication is done through UART connect and with the aid of a control line.

21.6.3.3 Implementation

This board is located on the server and in fact is the sever itself.

21.6.4 74HC7541

21.6.4.1 Specs

Operates at 2 to 5 volts

It is a non-inverting Schmitt Trigger

Capable of buffering up to 8 separate signals

21.6.4.2 Purpose

This chip is utilized to buffer the communication lines between the controller node and the sensor nodes. Without these buffers, communication is still possible between the chips, but it is completely unreliable.

21.6.4.3 Implementation

These buffers are placed on every sensor node. There is also a buffer placed on the controller node, but is only used to connect to the sensor node.

21.6.5 74HC138

21.6.5.1 Specs

Operates at 2 to 5 volts

Converts a binary to a decimal up to 8 different outputs and 3 inputs

Capable of producing a positive value or a zero value

21.6.5.2 Purpose

This chip is used to control the chip select pin on the sensors.

21.6.5.3 Implementation

This chip is only used on the sensor nodes.

21.7 IMAGES OF FINAL PROJECT

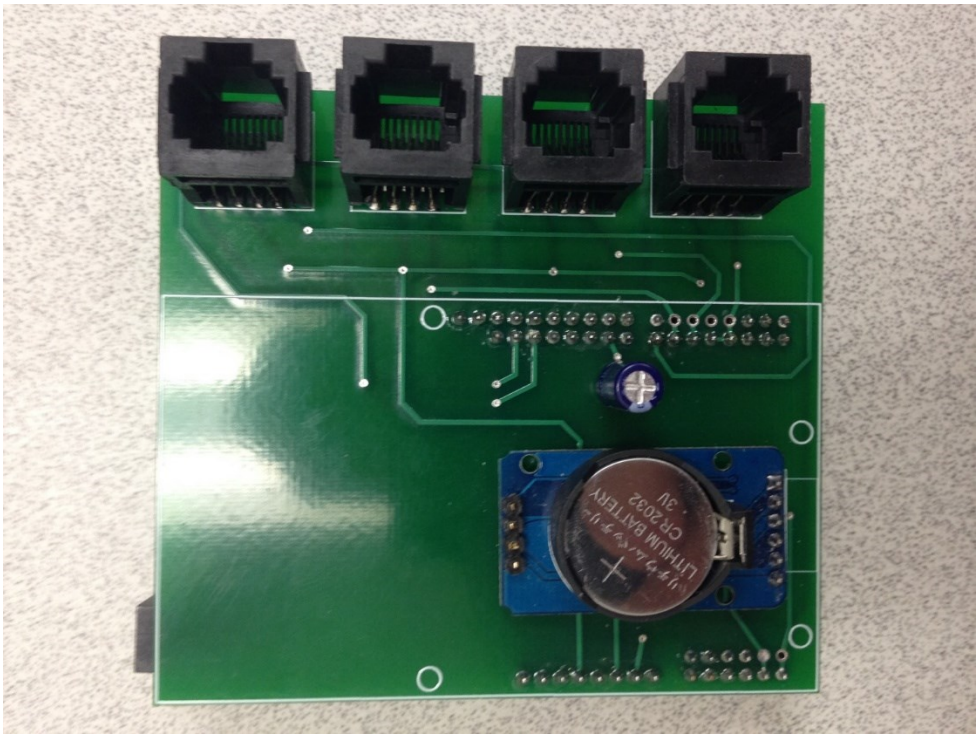


Figure 49: Sever PCB, Top View

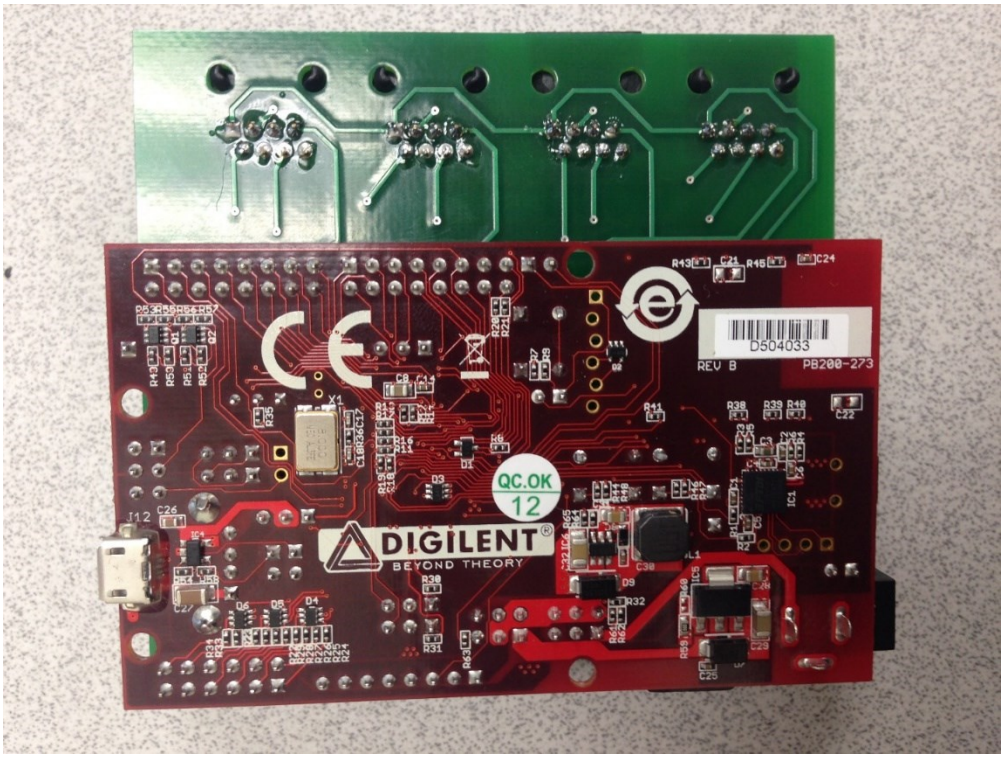


Figure 50: Sever PCB, Bottom View

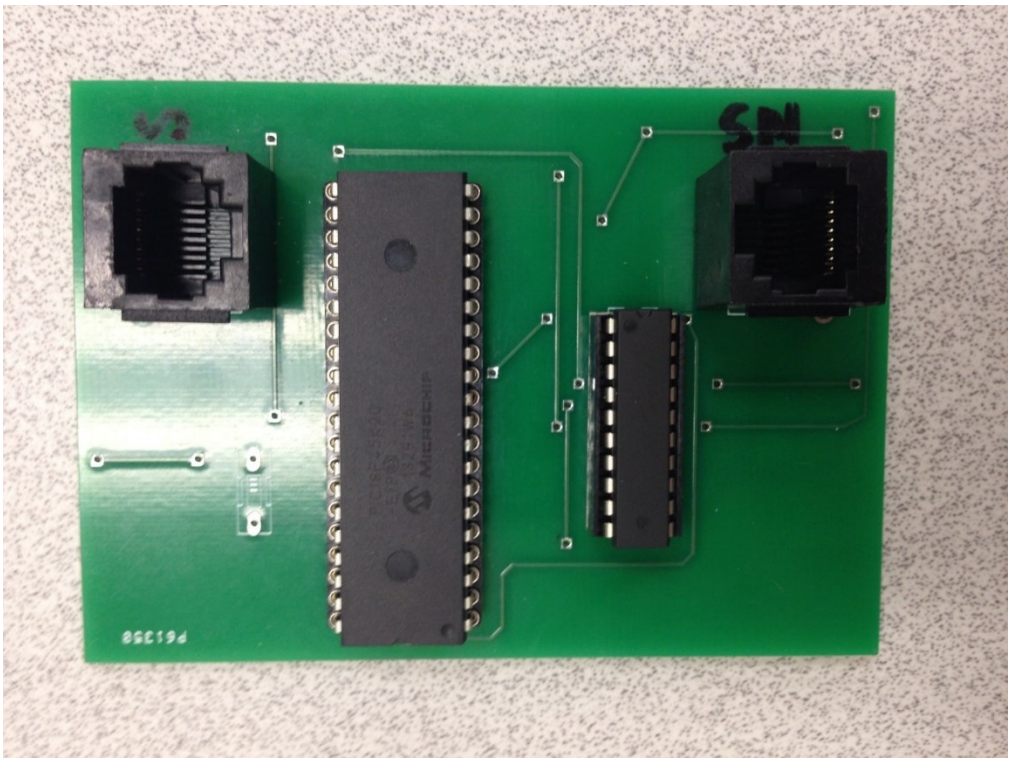


Figure 51: Controller Node PCB, Top View

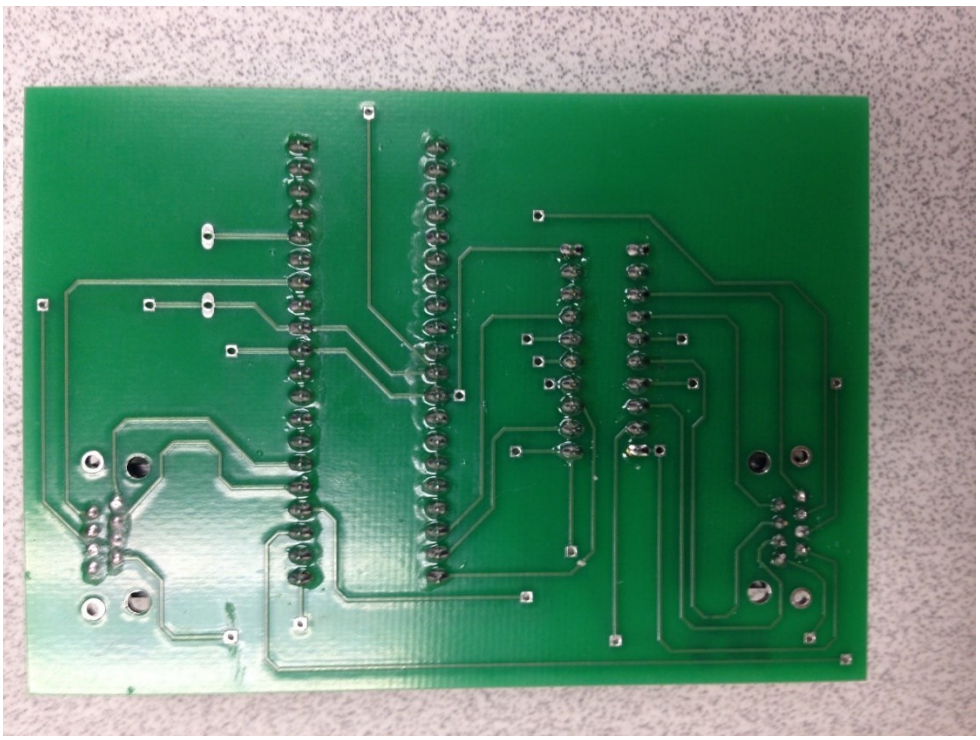


Figure 52: Controller Node PCB, Bottom View

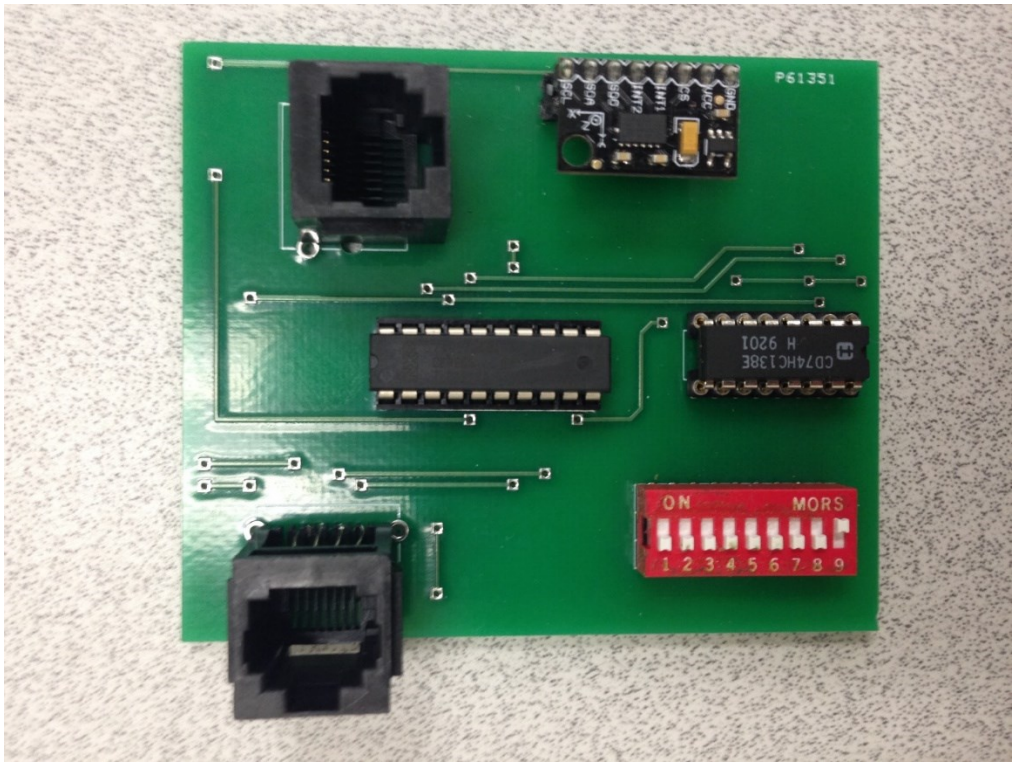


Figure 53: Sensor Node PCB, Top View

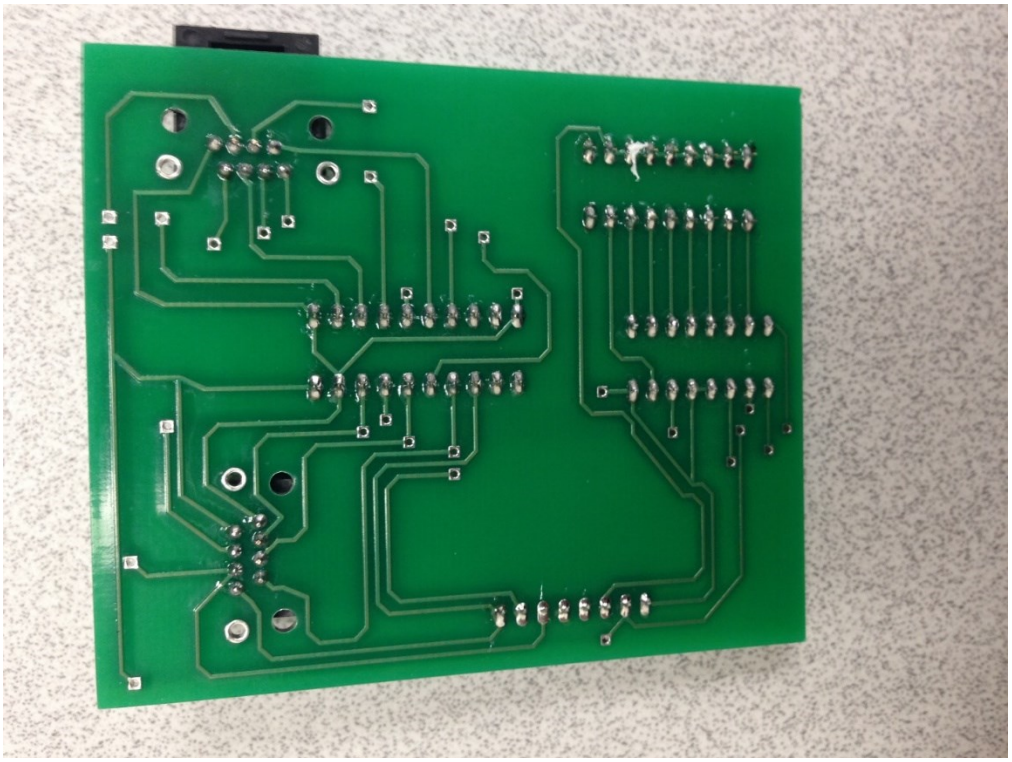


Figure 54: Sensor Node PCB, Bottom View

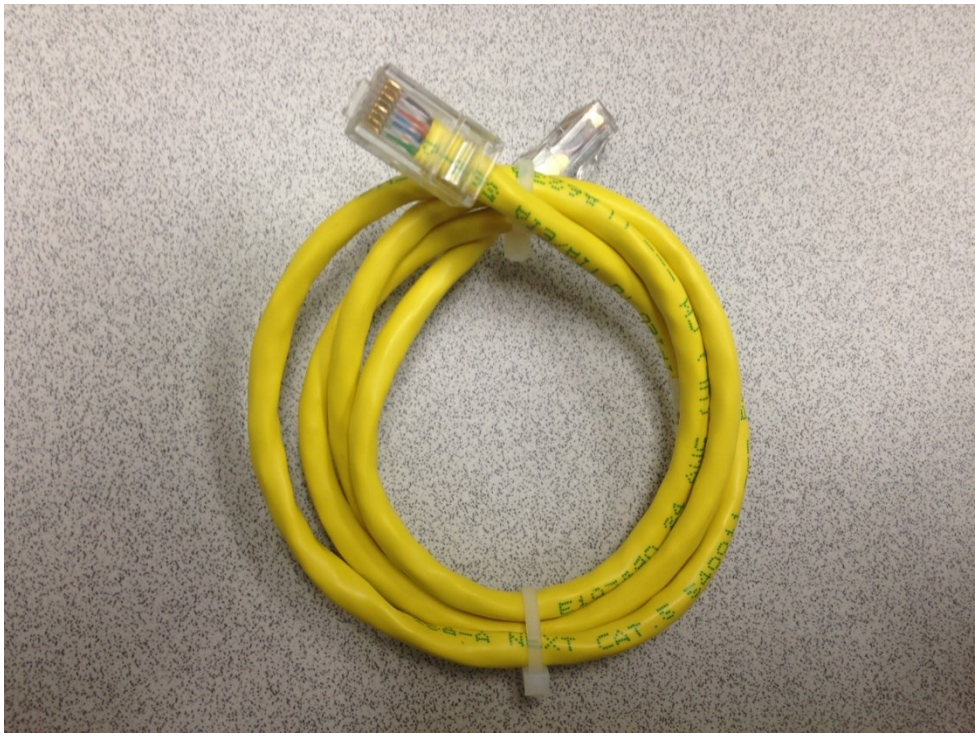


Figure 55: Straight Ethernet Cable

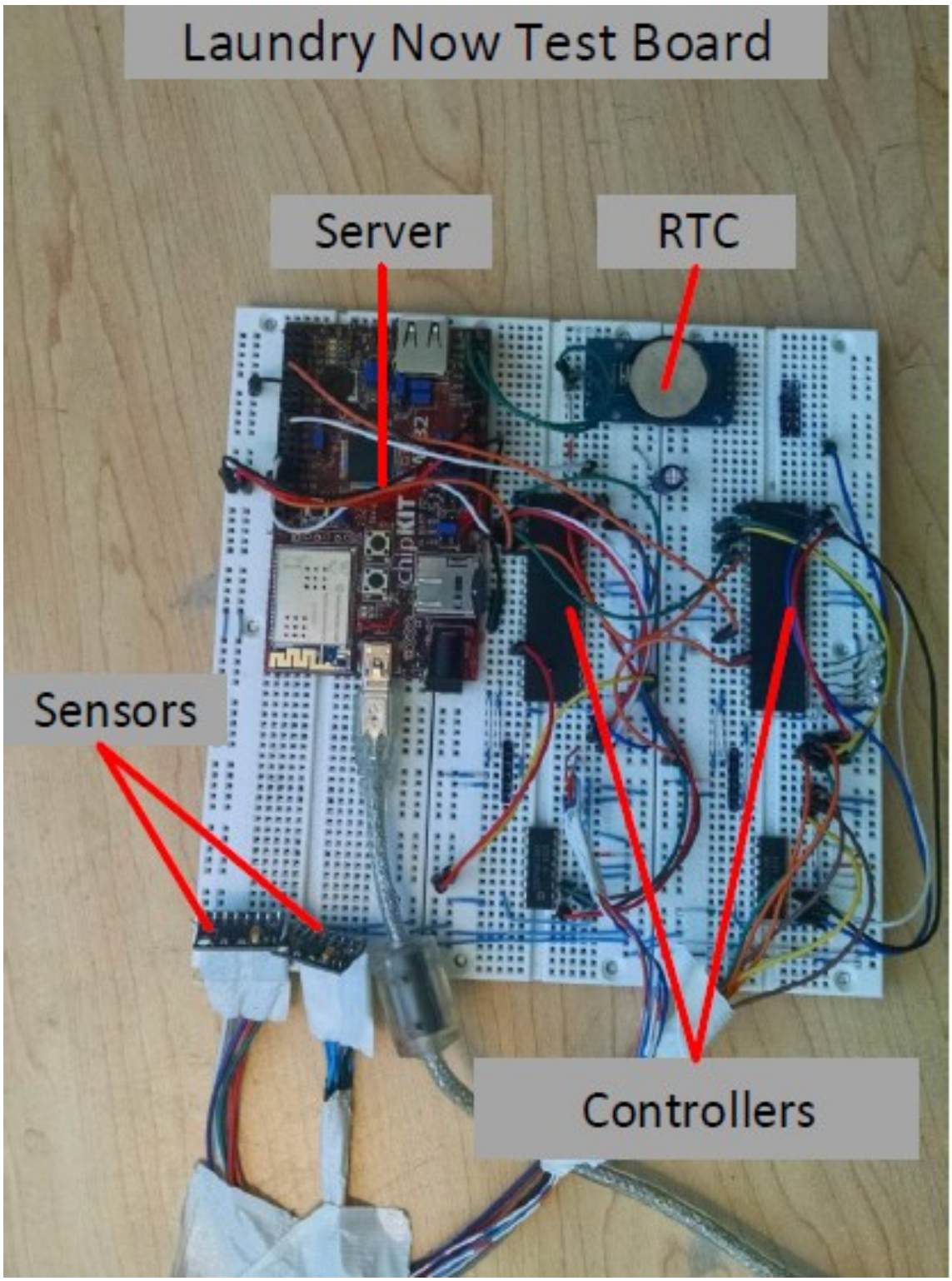


Figure 56: Test Board

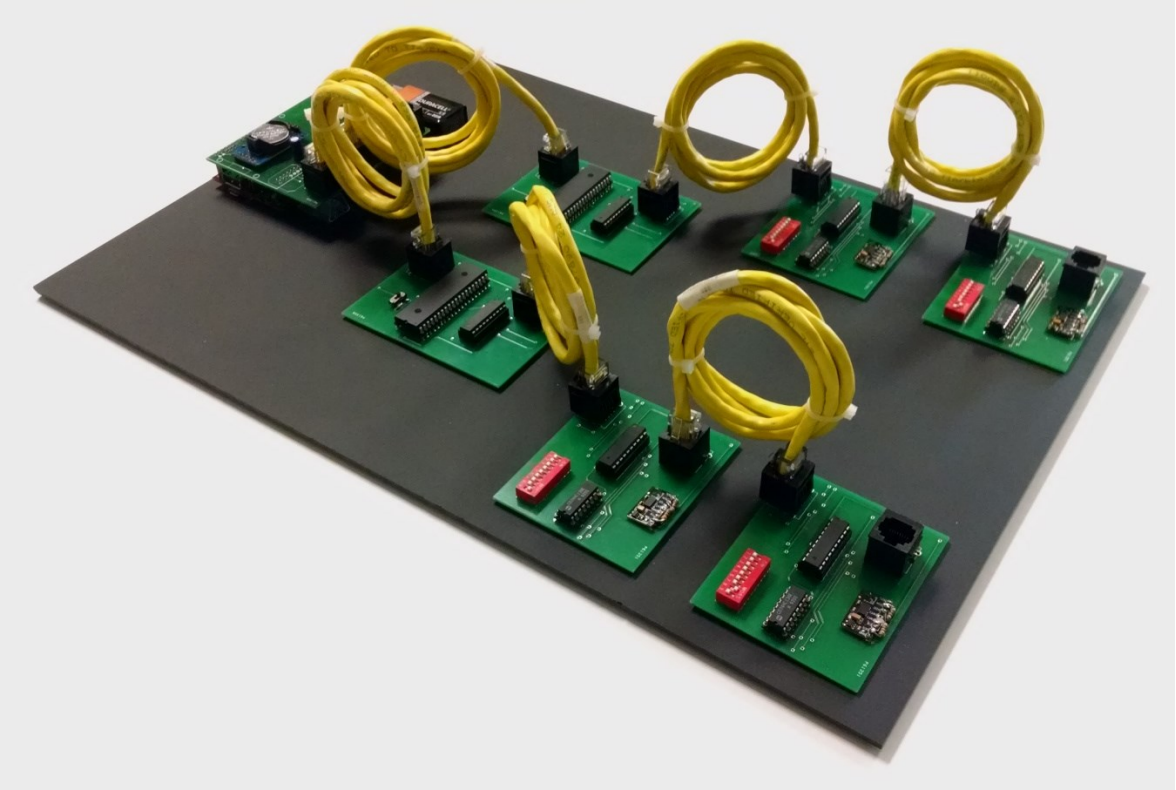


Figure 57: Final Product

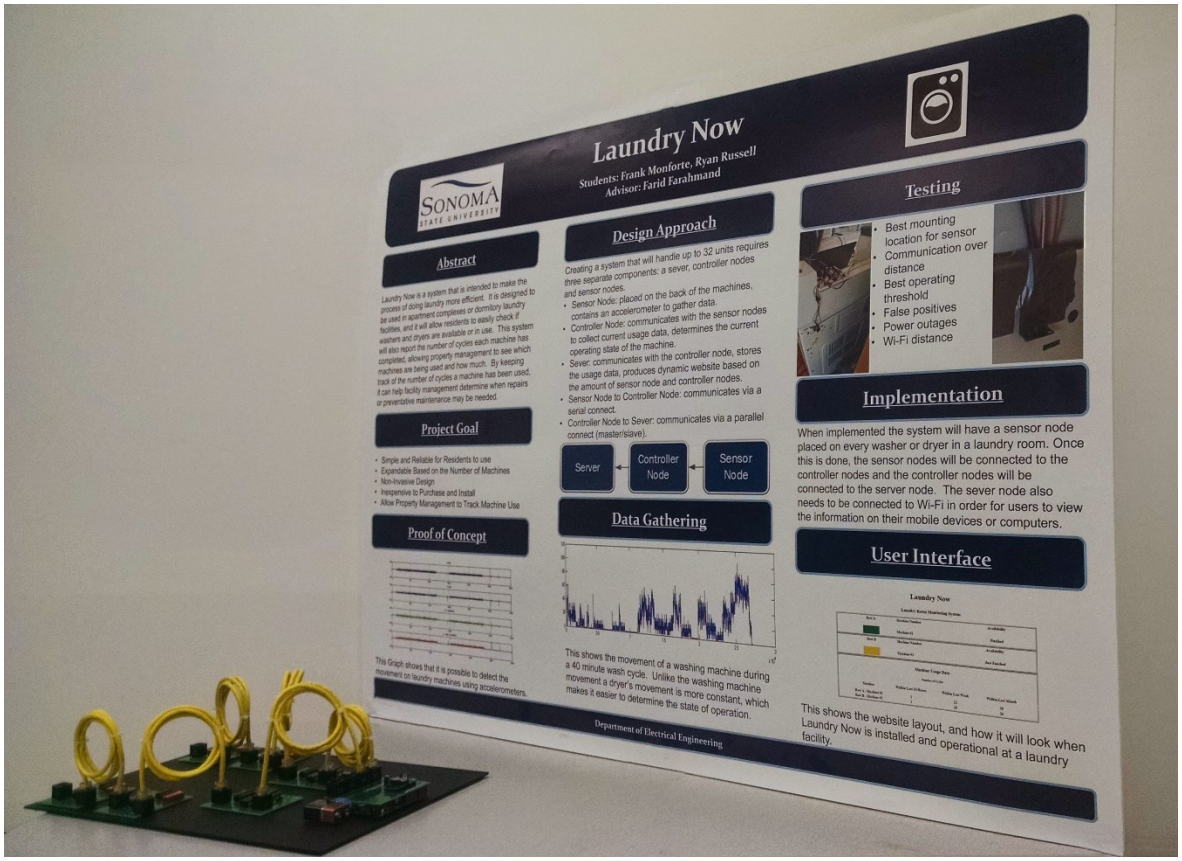


Figure 57: Final Poster and Project