WaterWatcher
by GreenFlow Technologies

CENG and ELEC 499

August 3, 2017

University
of Victoria

Group Information

S. No.	Name	V Number
1	Adam MacNeil	V00219572
2	Jean-Philippe D.Bouchard	V00241586
3	Josh Mehlenbacher	V00735261
4	Martin Kellinghusen	V00775537
5	Tritan Coombes	V00838936
6	Victor Weidemann	V00738950

Acknowledgment

GreenFlow Technology would like to thank the following people, without whom this project would not have succeeded:

Dr. Mihai Sima, our project supervisor;

The UVic IEEE Student Branch and the UVic ECE Department for their financial contributions;

PCBWay for the fabrication of our custom printed circuit boards;

CANAssist for allowing us to use their reflow ovens;

The ECE Lab technicians (Brent, Rob, Paul) for the valuable advice they gave us;

Ilamparithi Thirumarai Chelvan and Philipe Baback Alipour for the organizational aspect of the course;

Lastly, and most importantly, our family and loved ones who watched us work evenings and weekends this semester to get WaterWatcher to where it is today.

Contents

Executive Summary

GreenFlow Technologies has designed a real-time water flow meter for in home use in an effort to shift the water consumption behaviour in North America. The WaterWatcher is comprised of two units, one that connects to the water lines under the sink and the other that sits on the counter top, called Flow and Hub respectively. These units are designed to be both water and energy conscious, by displaying water consumption data to the user in a low power mode.

Both Flow and Hub use a low power microprocessor and the firmware is designed in such a way that they will be using next to no power when not in operation. As well as being designed for low power, Flow will be powered via a micro hydro generator connected inline with a flow meter. The two units are designed to use a low power Bluetooth module for wireless communication between each other.

Over the course of three months the team at GreenFlow Technologies have worked on WaterWatcher to bring it from a simple idea all the way to a proof of concept prototype. While there were some issues along the way, GreenFlow Technologies won third place at the University of Victoria ECE Capstone Project Demonstration in July 2017.

I Introduction

Canada is often thought of as a water provider for the world. Containing about 20% of the world's drinkable water supply, but less than half of that is renewable water [1]. In 2011, the average Canadian consumed 251 liters of water a day [2], yet the United Nations states a human being needs only 50L a day to survive [3]. This means that Canadians are using 5 times more than the minimum needed amount. GreenFlow Technologies has designed WaterWatcher to help the everyday consumer reduce the amount of water they use.

WaterWatcher is a water consumption measurement system that will display real-time water consumption data. With this live feedback, consumers become aware of their water consumption habits and are empowered to change their behaviour. WaterWatcher is therefore geared toward socially conscious users who have an interest in water conservation.

WaterWatcher has been designed to be easy to install, by placing one unit in line with each waterline of a sink, and another unit that is installed on the countertop. The units are called Flow and Hub respectively. Flow will measure the water volume flowing through the waterline, while Hub will display the water data to the consumer. Special consideration was put into the design of each unit to reduce total power required, thereby allowing the customer to be energy efficient as well as water efficient.

II Objectives

- GreenFlow' mission statement is to alter the North American water consumption paradigm, thereby reducing water usage and educating the public on better water practices.
- WaterWatcher has been designed to provide a real-time feedback loop to the user of the current water consumption while the tap is on.

III Design Specifications

i Overall System Requirements

The first prototype of WaterWatcher is meant to be installed under a kitchen sink. The first step that the GreenFlow team took was to find the average water flow rate for a standard kitchen sink. In the *Sink Flow Rate Tests* table found below, the range of max flow rate for a typical sink was found to be between 4-9 liters per minute. This means that the flow meter selected for the project needs to be able to handle at least a range of 4L/min to 9L/min. The flow meter selected for the project has a range of 1L/min to 30L/min, and is meant for ½" piping, which is one of the common North American kitchen sink connection sizes [4].

Table 1: Sink Flow Rate Tests

	Sink 1	**Sink 2**	**Sink 3**	**Sink 4**
With aerator	4.65 L/min		4 L/min	4.8 L/min
Without aerator	6.66 L/min	9.3 L/min		

Being a resource conscious product, each unit of WaterWatcher was also designed for low power consumption. A low power microcontroller was selected for the project [5]. The STM32L053R8 was selected as it had the added benefits of being able to run at 32MHz, had all the required peripherals, and had development boards that the firmware team could use while the hardware was being developed.

Lastly, as WaterWatcher displays real-time water consumption data to the user, and has a goal of being easy to install, two separate units where designed to communicate between each other via Bluetooth to facilitate the installation. Bluetooth was selected as the transmission protocol between the units as it has very low power consumption and an appropriate data range compared to other methods of wireless transmission [6]. The Atmel ATBTLC1000-MR110CA BLE module was selected as it is a low power module [7].

ii Hardware Design

(i) Hub Design

The main purpose of Hub is to take the data it receives from Flow and display the information to the user in real-time. However, the main consideration in the design of Hub and Flow was to be low power. In that regard the microprocessor was selected due to its low power consumptions.

The power system for Hub was designed to be powered from a rechargeable battery. The battery selected was a single cell 18650 lithium-ion as it is rechargeable, has a high-energy density, and correct capacitance for the Hub design [8]. The battery chosen for the project has a nominal voltage of 3.7V, but has a usable range of 3.2V-4.2V [8]. Given that the system is designed to operate at 3.3V, it was necessary to use a buck-boost converter to be able to maintain the correct system voltage. The Texas Instruments TPS63031DSK voltage regulator was chosen because it is a buck-boost regulator, has a fixed 3.3V output, and is high efficiency [9].

To display the data to the user a four row 20-digit LCD screen was selected. It was selected as it provided more information to the user than a standard two row 16-digit display and required less effort to implement than a high-resolution screen. The added benefit of the selected LCD is that it is powered via the same 3.3V as the rest of the board and requires not additional voltage circuitry [10]. On top for the LCD display there are 4 push buttons implemented in the Hub design for LCD menu control.

(ii) Flow Design

The Flow unit's main purpose is to measure the volume of water flowing to the sink and send that data to the Hub unit. The microprocessor for Flow is the same as for Hub, this was done to ease the firmware development, and all the benefits of the microprocessor apply to Flow. Added to the low power microcontroller selected for both units, the Flow was designed to be powered via a micro hydro generator connected inline with the flow meter. As the microcontroller selected has a maximum input voltage of 3.6V [5], the generator that was selected has a regulated output and matching inlet size for the selected flow meter [11] [12].

The generator that was selected had very limited documentation available, so the decision was made to include the same battery and charging circuitry on the Flow unit that is on Hub, with the hope that it would not be needed and could be removed in later iterations. This addition proved to be a wise choice as the generator requires the flow rate to be above a certain level before it will provide a regulated output. Both the micro hydro generator and the flow meter selected are passive devices, which added to the security of the overall system as there is no chance a malfunction of the hardware would cause water damages to the home. With these components selected, the concept and preliminary design of the Flow is finished.

The block diagram for both Flow and Hub can be seen below in the figure, *Flow and Hub Block Diagram.*

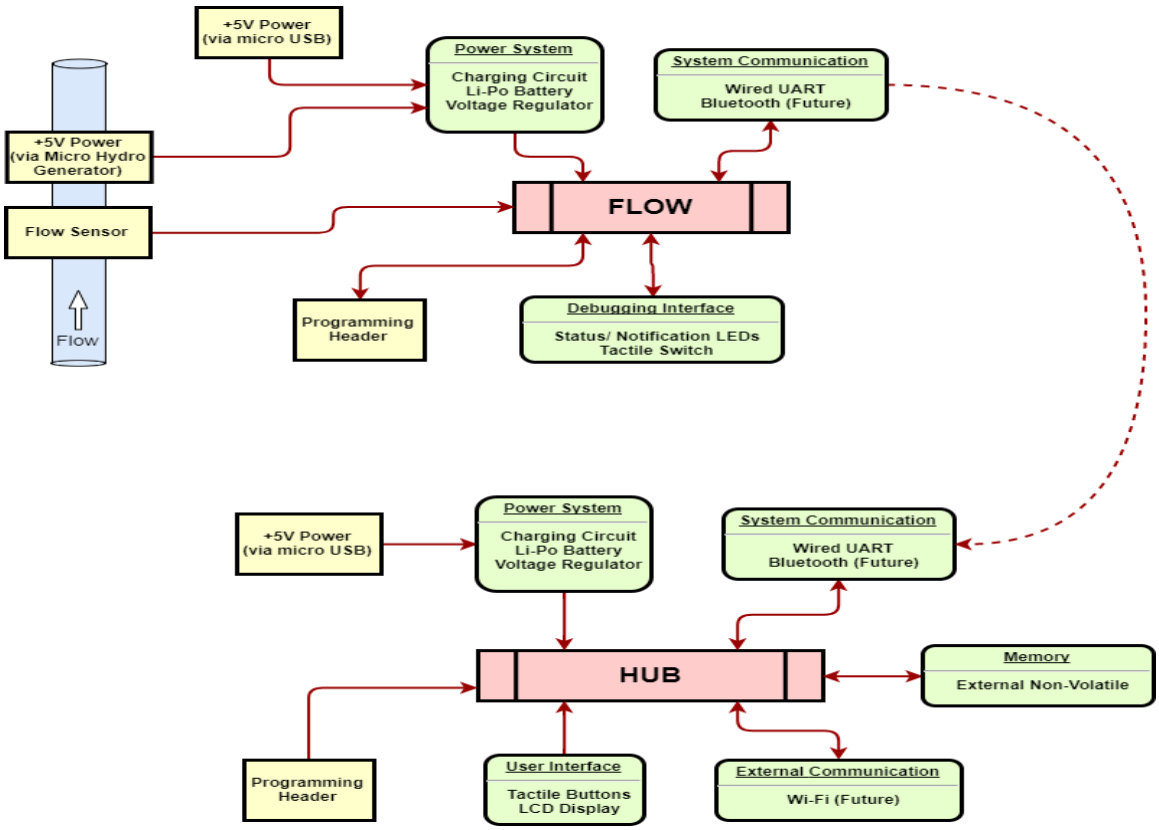

Figure 1: Flow and Hub Block Diagram

iii Firmware Design

As the main design consideration of both Flow and Hub was to be low power, the firmware team created state diagrams for each unit that would keep them in low power mode for as long as possible. As shown in the figures *Flow State Diagram* and *Hub State Diagram*, both Flow and Hub are in sleep mode while there is no flow of water through the flow meter or data being sent from one board to the other.

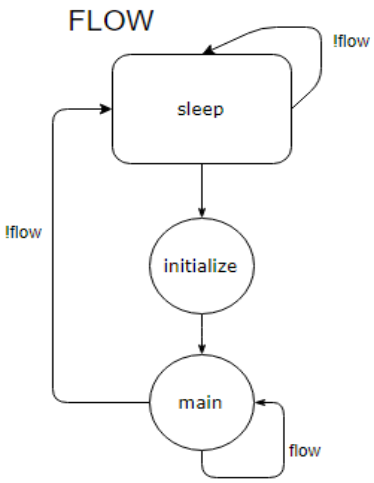

Figure 2: Flow State Diagram

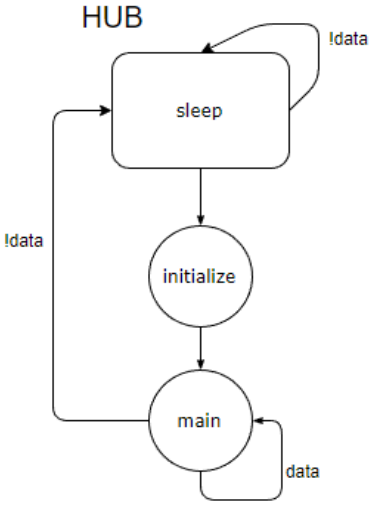

Figure 3: Hub State Diagram

As well as being power conscious, the firmware on Flow also needed to receive the data from the flow meter and transmit it to Hub. The output of the flow meter during test is seen in the figure *Flow Meter Output at 7.5 L/min* below.

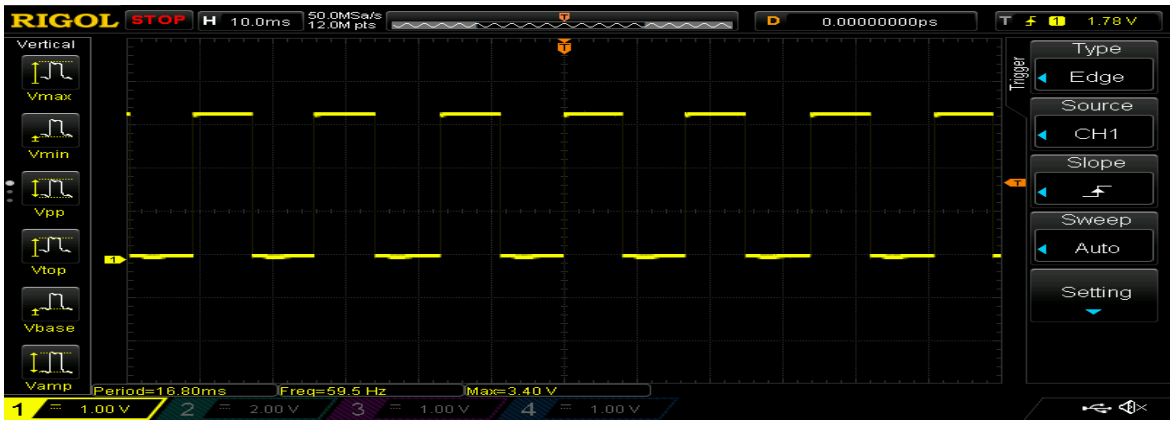

Figure 4: Flow Meter Output at 7.5L/min

To transmit the data from Flow to Hub, during development, UART with physical wires was used, and Bluetooth would be used once the hardware came in. The data from the flow meter, as well as battery status flags, and an error message are all sent from the Flow to the Hub in one data packet. The data structure is seen in the table *Data Structure* below. When transmitting the data, it is converted to a uint8_t array to be sent via the HAL library calls [13]. In the figure, *Data Transfer Between Units*, the data conversion from a data structure to a uint8_t array is shown while transmitting between units.

Table 2: Data Structure

Data Name	Data Type	Data Value
VolumeInTicks	uint32_t	Number of rising edges from flowmeter
flowChargerFlags	uint8_t	Flags from the charger chip
flowBatteryFlag	uint8_t	Flag if battery is below 3.3 volts
errorCode	uint8_t	Checks code to ensure packet sent successfully.

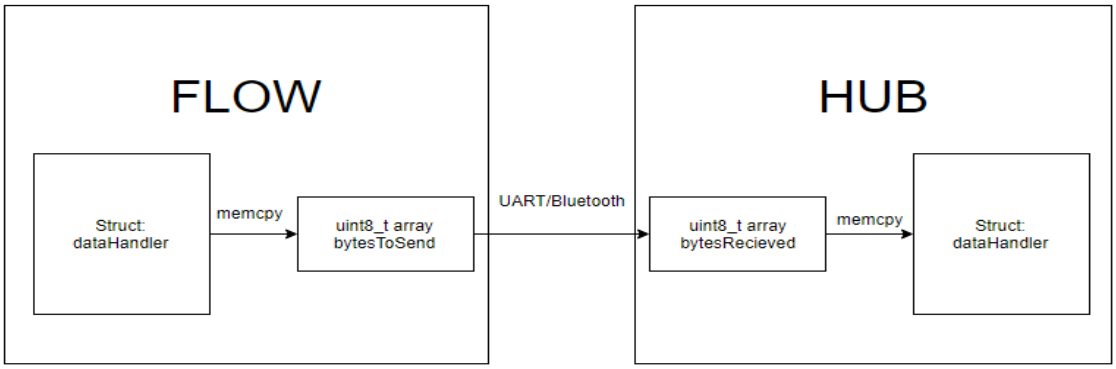

Figure 5: Data Transfer Between Units

IV Literature Survey

Currently, on the market there are different options for in home water flow meters. Solutions such as FLUID [14], Kamstrump [15], and even a DYI with a raspberry pi [16], all provide a variation on the same concept; a flow meter that connects to the main water line and provides data to a web or phone app after the water has been consumed. What WaterWatcher strives to do is provide real-time feedback to the user as they are consuming water, so that during their water use, they can make the conscious decision to reduce the volume of water used. In this respect, GreenFlow has an idea that is without competition in the market.

V Team Duties & Project Planning

i Separation of Duties

GreenFlow is comprised of six team members: three electrical engineering students and three computer engineering students. From the start of the project the tasks were divided along deliverable lines, with the electrical students in charge of the hardware, and the computer students in charge of the firmware.

The hardware team's duties were to design two low power units that would measure flow rates, communicate between themselves, and display data. In that respect, Martin Kellinghusen took on the design of the Hub, Adam MacNeil took on the design of the Flow, and Tristan Coombes took on both the power design of the Flow unit as well as the case design for both units.

The firmware team's duties revolved around taking the flow data, transmitting it between the units, and properly displaying the data on an LCD. To accomplish these three tasks Victor Wiedemann took on the role of firmware lead and coded most of the firmware for both units, Josh Mehlenbacher assisted in coding the Bluetooth connection between boards, and Jean-Philippe D.Bouchard tackled the task of displaying the data onto the LCD.

On top of both hardware and firmware, Jean-Philippe took on the role of Team Lead and organized team meetings, kept track of work logs, and wrote the reports; while Josh took on the role of website design.

ii Milestones

At the start of the project the GreenFlow team agreed to the milestone schedule seen below in the *Proposed Milestone Timeline*. However, due to circumstances explained in the Project Timeline section, the milestone delivery was as seen in the *Actual Milestone Timeline* figure below.

1. Research and Component Selection finished
2. Schematics Finished
3. PCBs Designed and Ordered
4. Firmware Finished
5. PCBs populated and tested
6. Code ported from dev boards to units.
7. Test complete

Figure 6: Proposed Milestone TimeLine

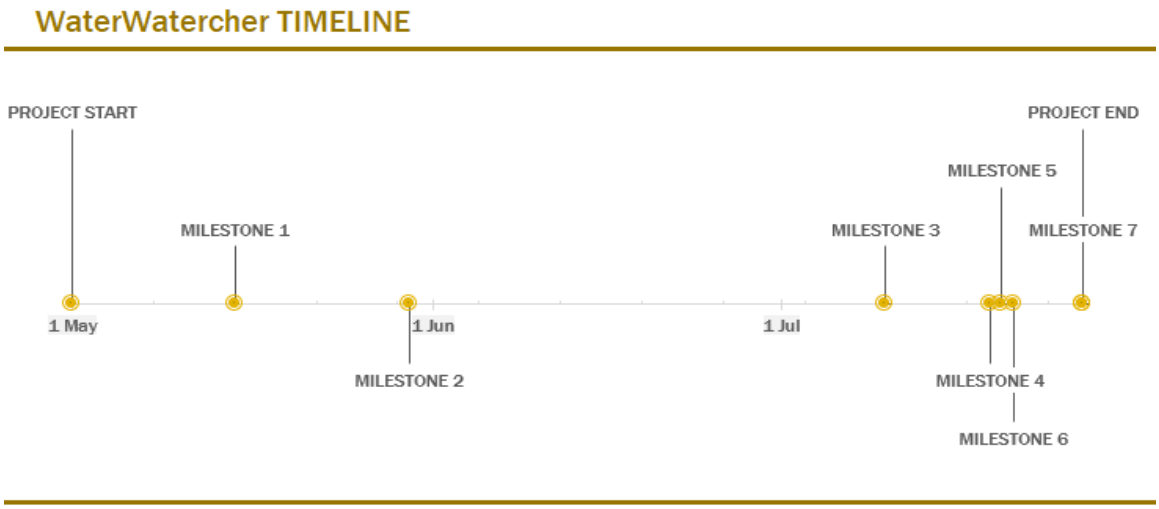

Figure 7: Actual Milestone Timeline

iii Project Timeline

The GreenFlow team spent approximately 550 hours working on WaterWatcher over three months. The breakdown of hours worked by task can be seen in the figure *Percent of Hours Worked by Task* below. Needing to create two independent units, the hardware team put in the most time in designing, assembling, and testing the hardware. The administrative tasks were comprised of designing the website, writing reports, preparing and following up on meetings.

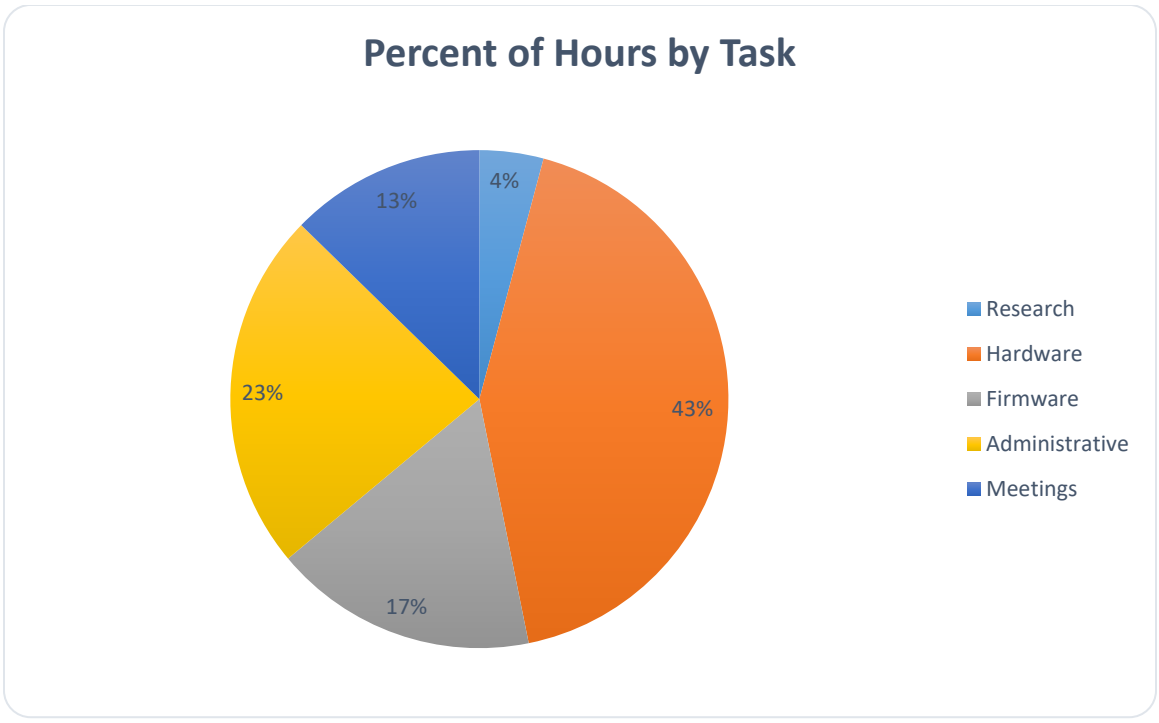

Figure 8: Percent of Hours Worked by Task

Halfway through the project, a member of the GreenFlow team went to an international competition for a few weeks. This was the main cause of the delay in milestone three and onwards. The result of this delay in the milestones can also be seen in the figure *Total Hours Worked vs Weeks*, below. Up until the return of the team member, everyone at GreenFlow was working consistently on WaterWatcher. Once the team member returned, the work progressed at an accelerated pace, in order to finish by the presentation day.

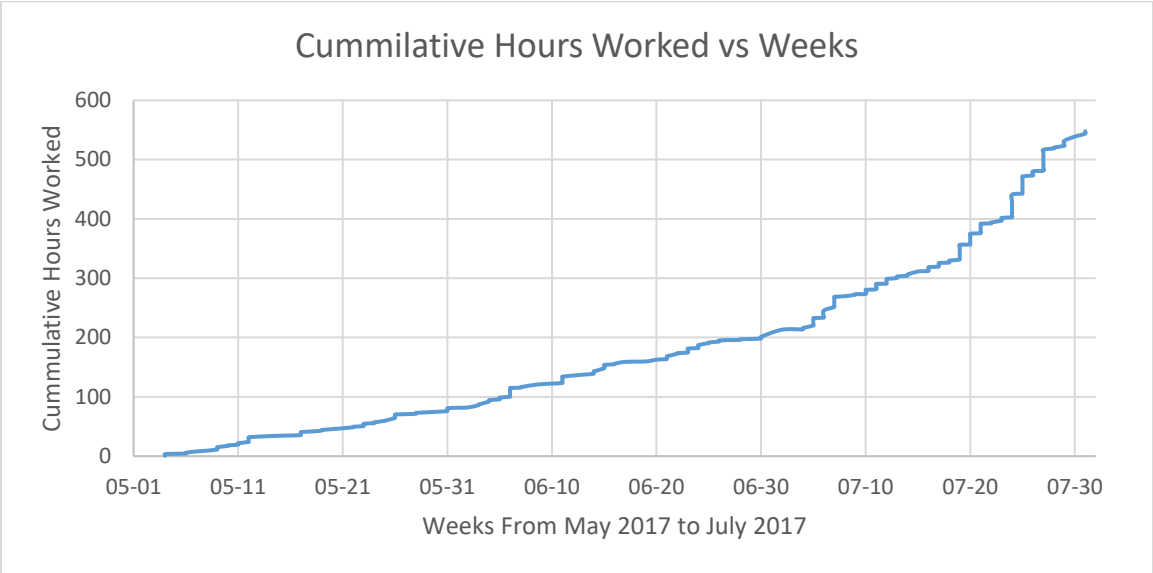

Figure 9: Total Hours Worked vs Weeks

VI Design Methodology & Analysis

Every member of the GreenFlow team understood that there was no time to spare in getting a prototype of WaterWatcher completed in a little over three months. Therefore, GreenFlow implemented the waterfall design model, as it is sequential and non-iterative. Once the team had agreed upon the minimum set of requirements, they were set in stone and the scope of the prototype never increased. During the design and prototyping of WaterWatcher there were some small changes made, but the scope of the requirements stayed constant. Once the prototyping was accomplished and the testing began, no drastic changes were made to the hardware. This methodology allowed GreenFlow to present a fully functioning prototype at the project demonstration.

VII Design & Prototype

i Hardware Design

Much of the hardware design was used in both Flow and Hub, making the job slightly easier for the hardware team. As discussed in the design specification section the STM32L0 was selected at it is a low power module. On top of that, in the battery monitoring circuit, high-side switches were added to the voltage divide for the ADC to avoid power loss when not reading the battery voltage.

The external crystal oscillator selected for both boards was selected to have a value of 7.3782MHz as this is integer divisible into the standard baud rates for UART and Bluetooth [17]. The load capacitors for the crystal were selected based off of equation 1. Given that the stray capacitance of the crystal is 2-5pf [18], and that the crystal capacitance is 12pf equation 1 can be rearranged into equation 2 to find the value of capacitors if they are set to the same value.

$$C_{xtal} = \frac{C_{L1} \times C_{L2}}{C_{L1} + C_{L2}} + C_{stray} \quad (1)$$

$$C_L = 2\left(C_{xtal} - C_{stray}\right) = 20pF \qquad (2)$$

Both Flow and Hub have three status LEDs used by firmware to indicate states. In normal experiment situations LEDs have a current limiting resistor to limit the current to 10-20mA. However, to conserve power whenever possible the current limiting resistors were selected to limit the current to 4mA.

The power input for Flow requires the option to be charged via the micro hydro generator or via USB. Since there are two methods to charge the unit blocking diodes were used to separate each charging method. The charge method is selected via a USBSET line which is pulled high for USB fast charge or pulled low for the micro hydro generator charge. The USBSET is connected to VUSB and is automatically put into fast charge when a USB is connected. The Hub contrary to the Flow only has the USB charging method implemented, and therefore, the USBSET line is always pulled high and connected to VUSB.

When implementing the Flow and Hub circuit designs onto the PCB certain considerations where taken. First off the regulator layout was based on the reference design found in the datasheet, as was the charge layout. The Bluetooth Chip was located as far away from the battery and regulator to avoid any possible interference. As Hub has the LCD positioned front and center on the PCB, the battery was position on the lower layer of the board to lower the PCB center of gravity and to simplify the assembly of components. Both the schematic diagrams and PCB layouts, as well as the bill of materials for both Flow and Hub are attached at the end of the report.

iii Firmware Design

Both Flow and Hub are running similar firmware. Yet it is in the specifications that each unit has key differences. The flow firmware takes the data and sends it to the hub, while the hub receives the flow data and displays it along with data unique to the hub. The firmware is explained in more detail in the following two sections.

(i) Flow Design

Given the state diagram for Flow shown in *Flow State Diagram* above both the initialize and main states are shown in depth below, in the *Flow – Initialize and Main State Diagram* figure. The initialize state will wake up Flow, initialize all the I/O and send a generic wake_up command to Hub before entering the main loop. The main loop waits for a flag timer, calculates the current volume of water, then builds the data packet and transmits it. The code does not send any data unless the flag has been set. The volume of water is calculated outside the main loop to avoid missing data. This loop continues as long as there is water flowing through the flow meter. When the flow meter stops incrementing the volume count, the reset flag gets set and the code puts Flow into sleep mode.

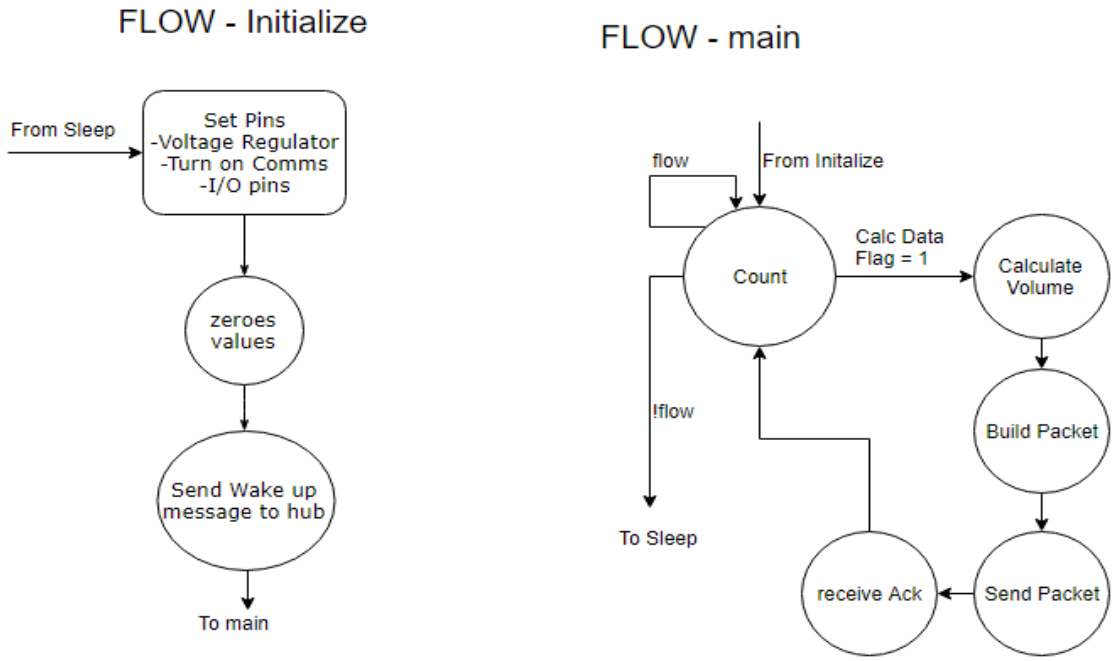

Figure 10: Flow Initialize and Main State Diagrams

The code structure for Flow is seen in the figure *Flow Structure* below. The *main* file only does the initialization and the loop that does the function calls to the other c files. The *flag timer* file checks to see if the UART or reset flags have been set. The *UART send* file calls functions from the 4 subfiles to build and send the data packet to Hub. The *battery check* verifies the voltage level of the lithium-ion battery and will flash a status LED and set the flowBatteryFlag to send to Hub. The *charger state* is similar to *battery check* except that the status LED turns on and does not flash, and the flowChargerFlag will be set. The status LED will flash to indicate low voltage if both the flowChargerFlag and flowBatteryFlag are set. The file *calculate volume* will count the number of rising edges from the flow meter since the last function call. Lastly the file *UART I/O* contains the code to transmit the data structure to Hub.

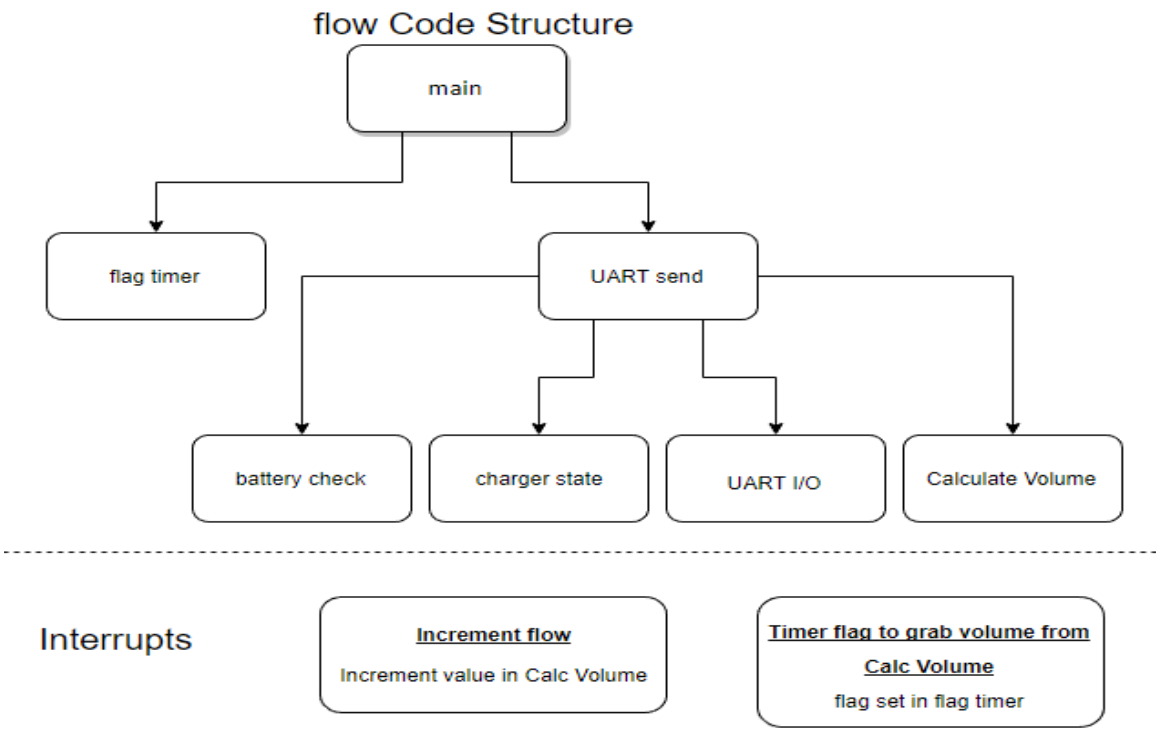

Figure 11: Flow Code Structure

(ii) Hub Design

Similar to Flow, the firmware for Hub has two main states which are seen in greater detail in the figure *Hub Initialize and Main State Diagram.* The initialize state for Hub is the same for Flow, other than it replies to Flow's wake_up command with an awake responce. The main code for Hub calls the update LCD function as long as there is data being sent from the Flow. The *Update LCD* function gets the data Hub received from flow by calling the grab data function. If an input button on the Hub has been pressed then the LCD will update accordingly.

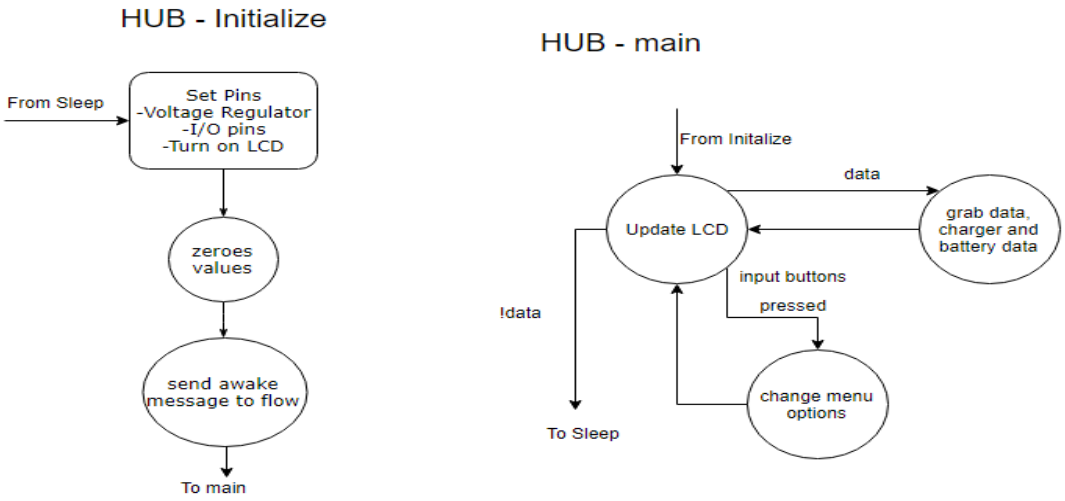

Figure 12: Hub Initialize and Main State Diagrams

The code structure for Hub is more complex than the code structure from Flow. On top of receiving data from Flow, it also has its own battery status data it can display. As with Flow, *main* just initializes and contains the main loop of the program. The file *flag timer* checks to see if the refresh LCD flag has been set. The *grab data* file grabs both the data that has been sent from the flow as well as the battery flags data. The file *battery check* verifies the voltage level of the lithium-ion battery and will flash a status LED and set the flowBatteryFlag to display the status message on the LCD. The file *charger state* is similar to *battery check* except that the status LED turns on and does not flash, and the flowChargerFlag will be set. If both battery flags are set on Hub then the LED will flash and the LCD will toggle between both status messages. The file *LCD control* takes the data struct passed to *main* from *grab data* and displays the information on the LCD screen. If a button has been pressed then *button states* returns the button information to LCD display and is handled accordingly.

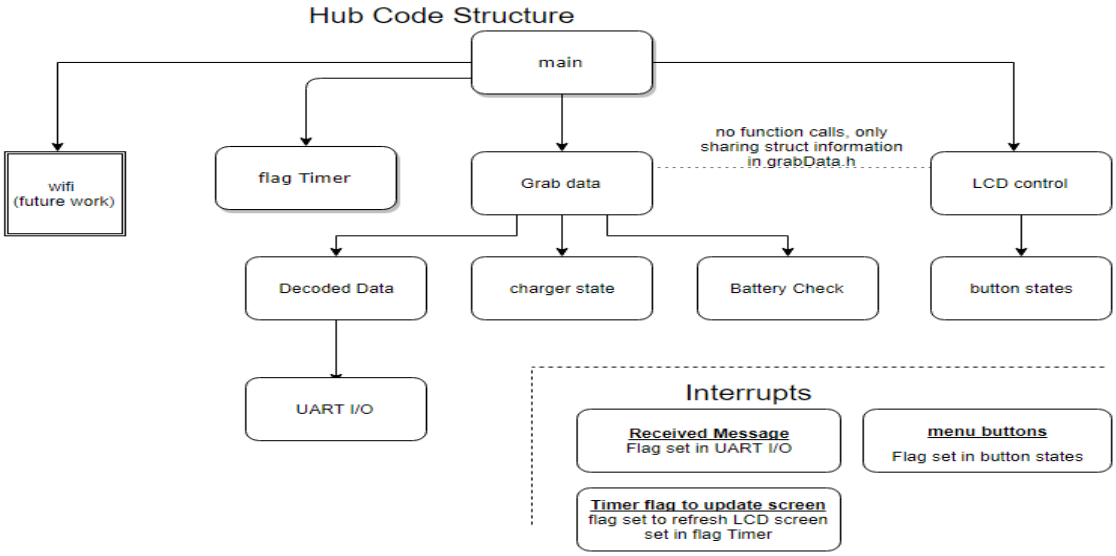

Figure 13: Hub Code Structure

(iii) Communication

The communication protocol set up during the development of the firmware was UART as it was the easiest to implement. However, for the MVP the communication protocol that would be implemented is Bluetooth. It was believed that the Bluetooth module selected would be able to be sent UART data, and transmit it wirelessly between modules. To avoid communication interference or hacking, a master slave piconet would mean that other slaves could not connect with each other in the network [6]. For the prototype a single master slave connection would be made.

However, due to a miscommunication between the hardware and firmware teams, the Bluetooth module selected was not able to receive UART data. This meant that a re-write of the ATMEL libraries would have been required to get Bluetooth up and running. This is something that the firmware team did not have enough time to do for demonstration day. Therefore, the prototype was communicating via UART and not Bluetooth.

iv Final Prototype

Due to the fact that the prototype at the demonstration did not have Bluetooth communication but UART, it was not the MVP of WaterWatcher. However, the GreenFlow team was able to demonstrate a real-time water flow meter with functioning sink. Everything but Bluetooth was functioning as envisioned in the early weeks of the project. Bluetooth will be part of the next iteration of the prototype, and part of the MVP.

v Next Iteration

As well as adding Bluetooth to the next iteration of code, during the development of the first prototype, certain recommendations came to light about the design that could improve the next iteration of the units.

Firstly the size of the units could be greatly reduced by using tighter routing and have a PCB with four layers instead of two. This would increase the cost of the individual units, but would make the installation easier. The component selection could be refined to lower the cost of the components. The battery was selected as an over estimation and with better power consumption data from the first prototype, a more appropriate battery selection could be made.

On top of these design changes that came up during the development of WaterWatcher, other recommendations occurred during the public demonstration. The batteries increase the cost and size of the unit, therefore removing them should be a priority within the next few iterations. Initially two methods where discussed, either have the microprocessor be able to do a fast boot and be powered solely from the micro hydro generator, or remove the microprocessor entirely and transmit raw water flow data from Flow to Hub via a method of communication that does not require much power such as radio frequency or Infrared. Each of these changes require drastic overhauls of firmware and hardware and would be implemented in the future after a few iterations of the current prototype.

VIII Testing & Validation

As there was little time left at the end of the development for testing, no concise testing plan was established for the prototype. The testing done was ad hoc, when possible. Detailed below, is the testing that was done on specific components, the finished hardware, the firmware, as well as on the system as a whole. Included at the end is a basic test plan if WaterWatcher would go into production.

i Component Testing

When ordering components for the WaterWatcher prototype, most had datasheets, however certain key components did not come with any documentation, namely the flow meter and micro hydro generator. Testing was done on these two components individually before they were integration into the system.

Testing for both the flow meter and generator comprised of filling up a 1L water bottle at different flow rates while measuring the outputs. While this method of testing is imprecise, it did manage to give rough estimates on values and confirm that the two components selected were appropriate for the task.

In the *Flow Meter Flow Rate Test Results* table, the results from the flow meter are seen. As the test was done to get a rough estimate of the accuracy of the flow meter, the large percentage in error rate between the measured rate and calculated rate is acceptable. If WaterWatcher was to be marketed, more precise testing would be required to get a low-cost high-precision flow meter. Another test done on the flow meter was to count the number of ticks the flow meter outputs when filing up 4 liters. This was done to verify that the firmware on Flow could take the output of the flow meter and not do any calculations to it. From the *Flow Meter Ticks Test Results* table, the number of ticks at faster flow was linear. Therefore, 463 ticks per liters was chosen by firmware. During the testing of the flow meter it became appearent that while at high flow rates the measurements are accurate, at lower rates the accuracy falls off. This could be rectified by finding a flow meter with greater precision.

Table 3: Flow Meter Flow Rate Test Results

Frequency [Hz]	**Time [s]**	**Flow Rate [L/min] (calc from time)**	**Flow Rate [L/min] (calc from freq)**	**% error**
4.42	84	0.714	0.589	17.493
16.7	25	2.400	2.227	7.222
59.5	7.9	7.595	7.933	4.456
99	4.76	12.605	13.200	4.720
164	3.1	19.355	21.867	12.978
172	2.9	20.690	22.933	10.844

Table 4: Flow Meter Ticks Test Result

Number of square waves	Time [s]	Flow Rate [L/min]
1816	200.35	1.2
1914	66.56	3.61
1904	18.72	12.82

When testing the micro hydro generator, both the flow rate and load resistance were testing variables, and both the output voltage and output current were measured. In the *Micro Hydro Generator Test Results* table, the minimum flow rates for each load impedance that would give a regulated output voltage from the micro hydro generator was found.

Table 5: Micro Hydro Generator Test Results

Load [Ω]	Time to fill 1L [s]	Flow Rate [L/min]	Voltage [V]	Current [mA]
30	3.12	19.23	5.04	163
40	4.5	13.33	5.08	124
50	4.98	12.05	4.96	99
60	5.57	10.77	5.08	83
70	5.7	10.53	5.04	70
80	6.09	9.86	5.04	63
90	6.15	9.76	5.04	55
100	6.65	9.02	5.04	49

ii Hardware Testing

Once both Flow and Hub were populated with components, a final visual inspection and first power up procedure were followed before beginning firmware testing. Before the visual inspection, a final cleaning of the boards using IPA to remove all flux residue was completed. Next using a microscope all pins were visually inspected primarily looking for short circuits, solder bridges, or improperly placed components. On Hub three solder bridges were identified during the visual inspection period. Using a hot air station the problematic sections were reworked until all solder bridge were corrected.

The next step before power up included short circuit detection using a multimeter with fine probes set to resistance mode. The resistance was measured between all major power lines on both Hub and Flow including Ground to +3.3V, VBAT, and VUSB. During the resistance testing there was a detection of a short circuit between ground and VUSB on Hub. The problem was detected on a solder bridge on the USB connector. Using a hand soldering iron the excess solder was removed and the resistance test was completed again.

Once all testing passed it was time to move on to first power up for both Flow and Hub. Using a current limited bench top power supply set to 4.2V at 100mA Flow and Hub

were both powered independently. During the first power up, current was monitored and would be cut off in the event of a fault. Both boards passed the first power up without any issues or shorts. The last step in the start-up sequence for both boards was to use a thermal imaging camera to monitor heat dissipation on both boards and verify that there were no hot regions present under normal operation condition. Upon successfully completing first power up of both boards the hardware and firmware integration testing began.

iii Firmware Testing

The firmware was developed on the STM32 Nucleo board, which allowed for debugging as the code was developed [19]. One of the main bugs found by firmware was during the UART transmission; only one of the two bytes of sent data was being received by the second board. The solution was to change the receive function to work via interrupt.

Another issue encountered with the firmware was the initialization of the LCD. Because of the way the shift register and LCD were wired in hardware, the data bits needed to be flipped before being sent to the shift register. Debugging was done with a Saleae Logic analyser allowing the visualisation of all the data bits and clocks being sent to the LCD, as seen in *LCD Debug* figure below. Another issue was that the microprocessor was sending the data to the shift register too quickly, and required additional delays when sending the data.

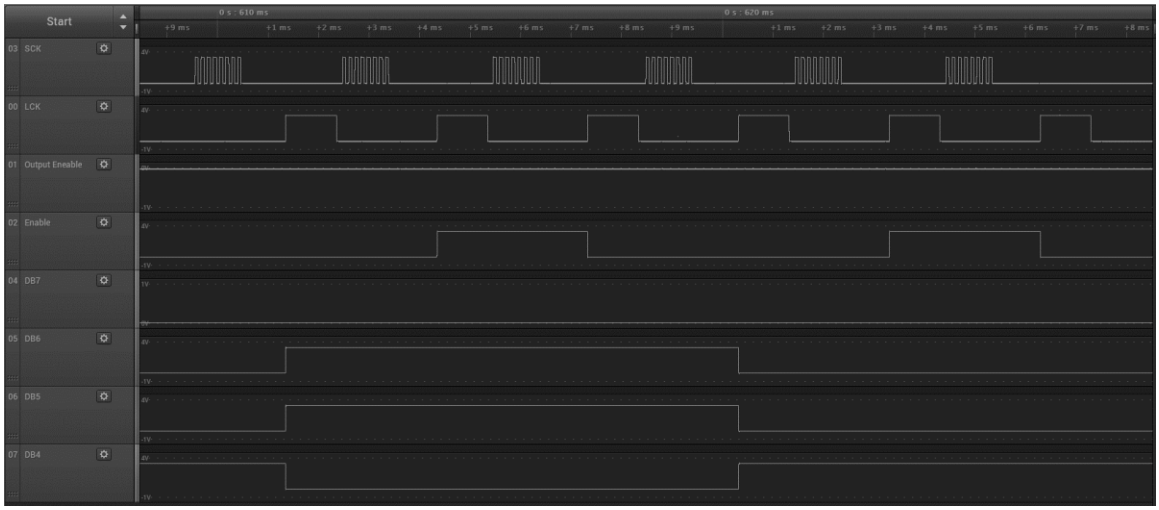

Figure 14: LCD Debug

iv System Level Testing

Once the firmware was loaded onto the hardware, a final check of all the components was done to verify that nothing else was missed in firmware. Verification that the LCD would turn on and initialize as expected, and that the LEDs and Switches were properly labeled in the code. The battery check flags were tested to verify would be set as expected and that the flow meter would interact properly with the hardware. Finally, the last test was to verify that the flow rate was being properly calculated on the hardware.

Two main issues were found during the system testing, the crystal oscillator was not being initialized and the boot0 pin was not pulled low. The boot0 pin not being pulled low meant that the firmware would not run on it's own unless the debugger was running the program. This issue was difficult to track down and require hours of debugging and research. While the microprocessor was using an internal crystal oscillator. The firmware was changed to use the more precise external oscillator.

v Manufacturing Test Plan

If WaterWatcher were to go into manufacturing, the testing could not be done ad hoc like it was for the prototype. A full test plan from the key components, to the hardware, to the complete system would be needed. As the goal of GreenFlow is to be water and energy conscious, thorough testing off all components would be needed to verify that no unit that leaves manufacturing is faulty.

Firstly, benchmark values for both the flow meter and micro hydro generator would be needed, and they would all need to be tested against those values. The assembled hardware should be visually inspected and loaded with test firmware that would test all the features of the components to verify proper operation. Lastly, the units should be tested as a full system to verify that the system functions as intended.

IX Cost Analysis

To get the initial prototype of WaterWatcher ready for demonstration in three months, it cost just under $570 in parts and materials, and just over 550 hours of work. However, the cost of components and parts for a single Flow and Hub is just under $200. This cost could be reduced in large part by placing bulk orders on the required components and parts. For the purpose of the prototype one of each part and the minimum quantity of components needed were ordered. This increased the final cost of the prototype, but means that in production the units would be significantly less expensive.

Another method of reducing the cost of WaterWatcher to the end user is to look at different methods of capturing and displaying data. Currently the flow meter and LCD screen are the two most expensive components of WaterWatcher. For capturing the data a different flow meter that is more massed produced could help reduce the cost. As for displaying the data, the hub could be integrated into a smart home device in the future, thereby removing the need for a display all together.

Due to the fact that the goal of the product is to reduce personal water consumptions, it is the belief of the GreenFlow team that local governments could provide grants to residents in purchasing WaterWatcher. Lowering the consumer's water usage would alleviate the pressures placed on the water systems of the cities. These government grants could further reduce the cost of WaterWatcher and therefore make it affordable for everyone.

X Conclusion & Recommendations

GreenFlow has taken three months to design WaterWatcher in the hopes of reducing personal water consumption in North America. Two units have been designed from scratch to be installed under and on the sink, called Flow and Hub respectively. Flow monitors the water flowing into the sink and transmits that data to Hub, which displays it to the user in real-time. This instantaneous feedback loop offers the user the ability to reduce their water consumption.

Working Over 520 hours, the team at GreenFlow Technologies delivered a first prototype just short of their envisioned MVP, missing only Bluetooth communication between the units. Flow measures the current volume of water being used in the sink and sends that data to Hub, which then displays the data real time to the user. Both units have successfully been designed to have low power consumption, both in the hardware component selection and in the firmware implementation.

The next steps for WaterWatcher would be to get the Bluetooth implemented to finish the MVP. Then GreenFlow would look at whether there is a better method of communication between both units that would reduce the power requirements of Flow. The next iteration of WaterWatcher after the MVP is complete would have a reduced form factor of both units, use a flow meter that functions at lower volumes, and lower the overall cost of the product. The next iteration of WaterWatcher will be a drastic improvement on the current one in terms of cost, size, and accuracy.

References

[1] "Frequently Asked Questions," Environment and Climate Change Canada, 16 02 2012. [Online]. Available: https://www.ec.gc.ca/eau-water/default.asp?lang=En&n=1C100657-1. [Accessed 07 2017].

[2] "Residential Water Use in Canada," Environment and Climate Change Canada, [Online]. Available: http://www.ec.gc.ca/indicateurs-indicators/default.asp?lang=en&n=7E808512-1. [Accessed 07 2017].

[3] "Water Consumption," Institute Water for Africa, [Online]. Available: https://www.water-for-africa.org/en/water-consumption.html. [Accessed 07 2017].

[4] "flexible supply lines," Home Depo, [Online]. Available: http://www.homedepot.com/c/supply_lines_valves_and_connectors_buying_guide_HT_BG_PL. [Accessed 05 2017].

[5] "ST," ARM Developer, [Online]. Available: http://www.st.com/en/microcontrollers/stm32l053r8.html. [Accessed 07 2017].

[6] "Bluetooth Basics," Sparkfun, [Online]. Available: https://learn.sparkfun.com/tutorials/bluetooth-basics . [Accessed 07 2017].

[7] "Atmel," [Online]. Available: http://www.atmel.com/Images/Atmel-42514-ATBTLC1000-MR110CA-BLE-Module_Datasheet.pdf. [Accessed 06 2017].

[8] "EBL 4 Pack 3000mAh 3.7V 18650 Li-ion Rechargeable Batteries 1200 Cycles," Amazon, [Online]. Available: https://www.amazon.ca/dp/B01E0SWQYK/ref=pe_3034960_236394800_TE_dp_1. [Accessed 07 2017].

[9] "TPS63031DSK Texas Instruments Switching Voltage Regulators," Mouse Electronics, [Online]. Available: http://www.ti.com/lit/ds/symlink/tps63030.pdf. [Accessed 05 2017].

[10] "New Haven Display," [Online]. Available: https://www.newhavendisplay.com/specs/NHD-0420H1Z-FSW-GBW-33V3.pdf. [Accessed 06 2017].

[11] "DC 0-80V/5V Micro Hydro Generator Tap 10W Water Turbine Generator Charging Tools Water Flow Hydraulic DIY," AliExpress, [Online]. Available: https://www.aliexpress.com/item/DC-0-80v-5v-Generator-10W-Water-Turbine-Generator-Charging-Tools-Free-Shipping/32776403568.html?spm=2114.search0304.4.1.SGA6B9. [Accessed 05 2017].

[12] "Black Hall Effect Water Flow Meter 1-30L/M G1/2" Thread 3-24VDC," Amazon, [Online]. Available: https://www.amazon.com/Black-Effect-Water-Thread-3-24VDC/dp/B00HG7HU30. [Accessed 05 2017].

[13] "STM32L4xx_HAL_Driver," ARMmbed, [Online]. Available: https://developer.mbed.org/users/EricLew/code/STM32L4xx_HAL_Driver/docs/tip/. [Accessed 07 2017].

[14] "Techcrunch," [Online]. Available: https://techcrunch.com/2015/09/15/fluid-is-a-smart-water-meter-for-your-home/.

[15] "MULTICAL® 21/ flowIQ® 210X," Kamstrup, [Online]. Available: https://www.kamstrup.com/en-en/products-solutions/water-meters/residential-water-meter.

[16] "Build a Wireless Water Meter for Your Home," IEEE Spectrum, [Online]. Available: http://spectrum.ieee.org/geek-life/hands-on/build-a-wireless-water-meter-for-your-home.

[17] "why 7.37 MHz," microchip, [Online]. Available: http://www.microchip.com/forums/m442874.aspx. [Accessed 06 2017].

[18] "Choosing the Right Crystal and Caps for your Design," adafruit, [Online]. Available: https://blog.adafruit.com/2012/01/24/choosing-the-right-crystal-and-caps-for-your-design/. [Accessed 06 2017].

[19] "Nucleo-L053R8," ST, [Online]. Available: http://www.st.com/content/st_com/en/products/evaluation-tools/product-evaluation-tools/mcu-eval-tools/stm32-mcu-eval-tools/stm32-mcu-nucleo/nucleo-l053r8.html. [Accessed 05 2017].

Glossary

uint8_t	8 bit data value
uint32_t	32 bit data value
UART	Universal Asynchronous Receiver Transmitter
Data Packet	data formatted for transmission
I/O	Inputs/Outputs
LCD	Liquid Crystal Display
Baud Rate	rate of information transferred in a communication channel
HAL	Hardware Annotation Library
MVP	Minimum Viable Product
Piconet	Ad hoc network that links wireless Bluetooth communication protocols
IPA	Isopropyl Alcohol