Kansas State University
Electrical and Computer Engineering
Senior Design

Ardent Mills Capacitive Flow Meter Final Report

Team 4

May 14, 2021

Luis Amiel
Meghan Clark
Jenalee Dickson
Valeria Morinigo
Joshua Richards

Executive Summary

Our project will assist Ardent Mills in creating a cheaper, more efficient flow meter. This product will attempt to replace their current infrared meters. Ardent Mills needs a flow meter to measure the flow of grains through their pipes, output the data received into a graph, and alert when there is too little grain, too much grain or no grain at all flowing through the pipe. These requirements will be met with our project. Our meter will measure the flow of solid particles through the spout in a mill. There are existing products that perform a similar function to ours, but we will implement additional features, such as alerting capabilities on an interactive graphical display, all at a very low cost/budget. Our finished product will be a functional yet affordable design.

Our report presents the design for a flow meter and the creation of this design. We show how it will work and how we built it. We first give a detailed explanation of the background and motivation for our project which leads to the goal, requirements, and the testing that is required for this project. Then we look at the project description, the technical design which includes both hardware and software design, and the design solutions we have created along with alternates that have been thought through. We address any global issues that we would see facing while working with Ardent Mills and their factories nation-wide and internationally. Finally we break down the work that each group member did and show a detailed schedule of how we worked on this project for the required 15 weeks of the semester. In this report we present the design of the system and how we built and tested the concept.

Table of Contents

PROJECT DESCRIPTION

Background and Motivation

Ardent Mills is a leading flour supplier and grain innovator company whose vision is to be the trusted partner in nurturing its customers, consumers, and communities through innovative and nutritious grain-based solutions. With a network of over 40 flour mills and bakery-mix facilities throughout the U.S., Canada, and Puerto Rico, Ardent Mills offers the broadest range of premium multi-use flours, whole grains, mixes, and custom multi-grain blends.

One thing that makes Ardent Mills products unique is the process, the system was designed to preserve the integrity of the grains. To fulfill these expectations, facilities use different types of devices that allow the collection of relevant data for better product identity in the finished product. Flowmeter sensors provide the most accurate and repeatable flow measurements for a specific application, whether for process control, or general research activities.

Currently, Ardent Mills uses a Buhler NIR infrared meter which is expensive. Every plant also has multiple “in-line scales” that measure product quantity while also controlling the rate. An application for our flow meter would be to verify that the in-line scale is creating a product flow to the rate it is set to. In this manner, by setting a range of leniency for the meter we are going to be able to control the difference between the set rate on the NIR and the measured rate for the sensor, to set an alarm if the rate is too far apart. We would also like to add other specific tasks such as controlling what percentage of each grain is necessary for the mixing of products and the ability to alert when water is needed to keep the moisture at the desired level, based on our input of grains into the system. All this data will be collected into a file, creating a history of the meter, for future references.

Project Goal

This project will measure the flow rate of solid particles through grain pipes in a mill. The device will issue warnings if the flow rate exceeds a certain boundary, and this information will be output to a data file and also accessed through a touchscreen monitor.

Project Requirements

ID	Project Requirements	Description
1.0	Output an analog signal for the Arduino	**Primary Functional Requirement:** Ardent Mills wants to be able to see how much grain is being moved in the pipe. This will need to be outputted on a LED screen then to a document(ex. Excel or an HTML site). It needs to output a range of 0-5V for the Arduino's analog-to-digital converters.
2.0	Create an output document	**Sub Functional Requirement:** This device needs to be able to record what it has measured. This will then be stored for a certain period of time.
3.0	Size: Minimize	**Constraint:** The device cannot block the grain from flowing through the pipe, therefore the device will need to be designed so that it does not hinder the flow in the pipe that will be at a minimum of 2.5" diameter.
4.0	Power Supply	**Constraint:** These flow meters will either need to be at the ends of the pipe or somewhere in the middle meaning that they will need to have a battery or a plug in as the power supply.
5.0	Alerting Capabilities: Red/Flashing LED turned on	**Objective:** This device will need to have an alert function when there is too much, too little, or zero grain flowing through the pipe. There will be a different alerting signal for each of these cases.
6.0	Maintain a cost that would be reasonable	**Constraint:** This device will need to be something that isn't super costly so that can implement it in multiple mills throughout Ardent Mills.
7.0	Accuracy	**Objective:** The flow meter will need to detect how much grain is flowing through the pipe within a certain accuracy rate.

Validation and Acceptance Tests

Ardent Mills needs this device to be able to read the amount of grains flowing through the 2.5" to 8" pipe, output and record this data, and alert them when too little grain or no grain is flowing through the pipe. These will all need to be working successfully with our design. We plan on successfully executing this with the requirements below. We will be testing these requirements with popcorn kernels and a pipe to simulate the grain following through the pipe. We will send the popcorn kernels down at different rates to make sure that the flow meter can accurately detect what is occurring in the pipe. Other design requirements that will need to be implemented and reviewed can be seen below.

Verification Matrix

Requirement ID	**Requirement**	**Requirement Verification Method**			
		Similarity	**Review of Design**	**Analysis**	**Test**
1	Output signal is 2.5V			X	
2	Differentiate between zero flow and non-zero flow			X	
3	Distinguish between flow rates				X
4	Alerts of zero flow, low flow and high flow				X
5	Graph data coming in every 2 seconds			X	
6	Operate for a period of time without needing to be replaced	X			
7	Flow meter does not interfere with grain flow in the pipe		X		

Validation and Acceptance Tests Summary

To test the requirements above we will verify these works with the following steps. To test that the sensor is outputting up to 2.5V we will test this with an oscilloscope. We can test this by putting different objects like a human hand or bag of popcorn down the pipe and making sure the oscilloscope can detect that something is being put in the pipe. Next, we can test that the sensors are reading correctly by pouring popcorn kernels down the pipe at a certain rate, pretty slow, and making sure it reads the correct rate even when there are no kernels flowing through the pipe, this will be done by calibrating the circuit with the kernels. This test would also make sure the circuit can detect what would be occurring in the mills with grain. To test the software we will need to make sure that there is a clear alert when there is no flow, low flow and high flow through the pipe. We also must graph this data on a readable screen and have the screen alert when the flow is inadequate. This data will then be stored. To make sure this design and product will be cost efficient for Ardent Mills we will use cost effective materials and make sure at a mass production our price will decrease. Once all of these things work separately we will put them together to make sure our entire flow meter works adequately. When all these tests are successful we will know our design works properly and will be efficient for Ardent Mills.

TECHNICAL DESIGN

Possible Solutions and Design Alternatives

Our solution will be to use a capacitive system by using two or more metal plates placed opposite each other around a non-conductive PVC pipe. We will have a circuit hooked up to one plate to power it and another circuit on the other plate to measure the difference in capacitance which will measure the flow of grain through the pipe. We will have a way to convert that signal into something our Arduino can read and output our flow rate through the pipe.

A possible design alternative for our flow meter could be an optical sensor. If we run into issues with a capacitive sensor we would next look at using optics. There would be issues with using optics because of the nature of most mills and the amount of dust that is produced by the different processes. If we were to use optics we would need to create a way to keep our sensors free of dust and debris to get an accurate reading.

Electrostatic flow detection is an alternative sensor technology, but we are concerned that introducing static electricity into the system could create a hazard of explosion.

If our all-electronic methods are unsuccessful, we could construct or purchase a paddle-type mechanical flow meter that would work with our Arduino to output the needed data for Ardent Mills. If we were to go with this option we would create a more extensive output system for display.

System-Level Overview

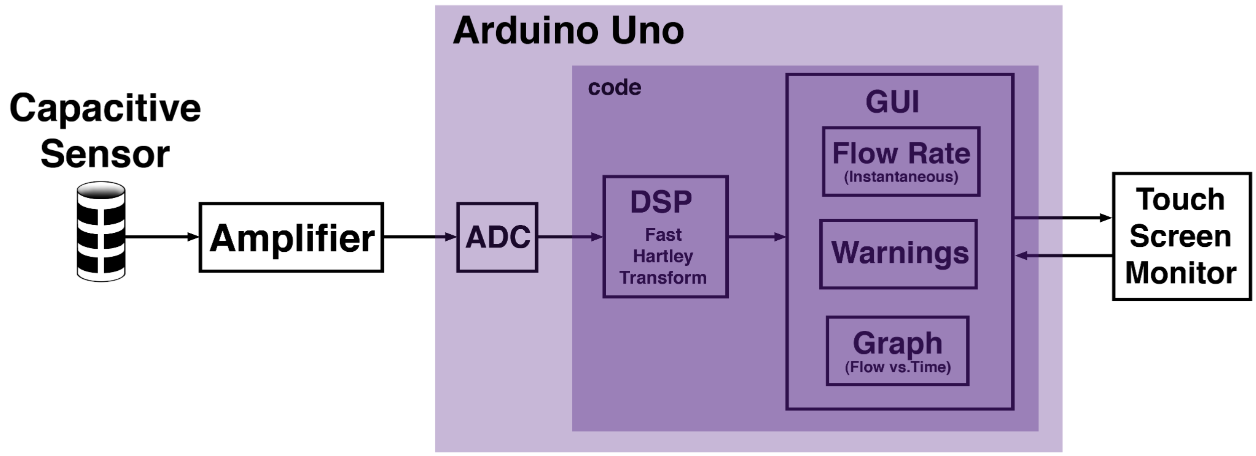


Fig. 1. Block Diagram of System.

Four cylindrical copper capacitors are fixed around a simple non-conductive plastic pipe. As grain flows through the pipe, the capacitance changes based on the mass of material. An amplifier will convert the changing capacitance into a varying voltage signal.

An Arduino Uno converts the varying voltage into the digital signal. The Arduino Uno will process and analyze the signal, and will control the touchscreen user interface.

Software

The Arduino ADC will convert the varying voltage into a digital number. Digital signal processing is used to extract the maximum frequency component of the signal, and the frequency will be converted into a flow rate, using digital signal processing. Flow rate data will then be saved to Arduino Uno's internal memory.

The Arduino Uno will control a graphic user interface to display three main items on a touch screen monitor: instantaneous flow rate, a graph of flow versus time, and system warnings of grain flow issues (this was not completed on our screen). Users will press touchscreen buttons or tabs to toggle between these different displays.

—–

Applicable Standards

The Arduino Uno is programmed in the C and C++ languages. Therefore, the C/C++ code must follow the International Organization for Standardization (ISO) requirements for those two specific languages: ISO/IEC 9899:2018, and ISO/IEC 14882:2020.

The Arduino Uno incorporates a USB B port, and it communicates with computers via the USB 2.0 standard. The last updates to the USB 2.0 standards were in 2014. [10]

Detailed Module Level Descriptions

Hardware Modules

Semi-cylindrical electrode capacitors

We will construct 4 semi-cylindrical electrode capacitors along a rigid PVC pipe. A "semi-cylindrical electrode" capacitor can be imagined as a parallel-plate capacitor, where each plate curves inward to form a semi-circular shape.

The electrodes will be strips of copper tape, wrapped into near semi-circles, around a PVC pipe. See Figure 2(a). We will cover the PVC with insulating tape, and then cover the electrodes with insulating tape, to prevent electrical connections between plate edges and to protect ourselves while testing.

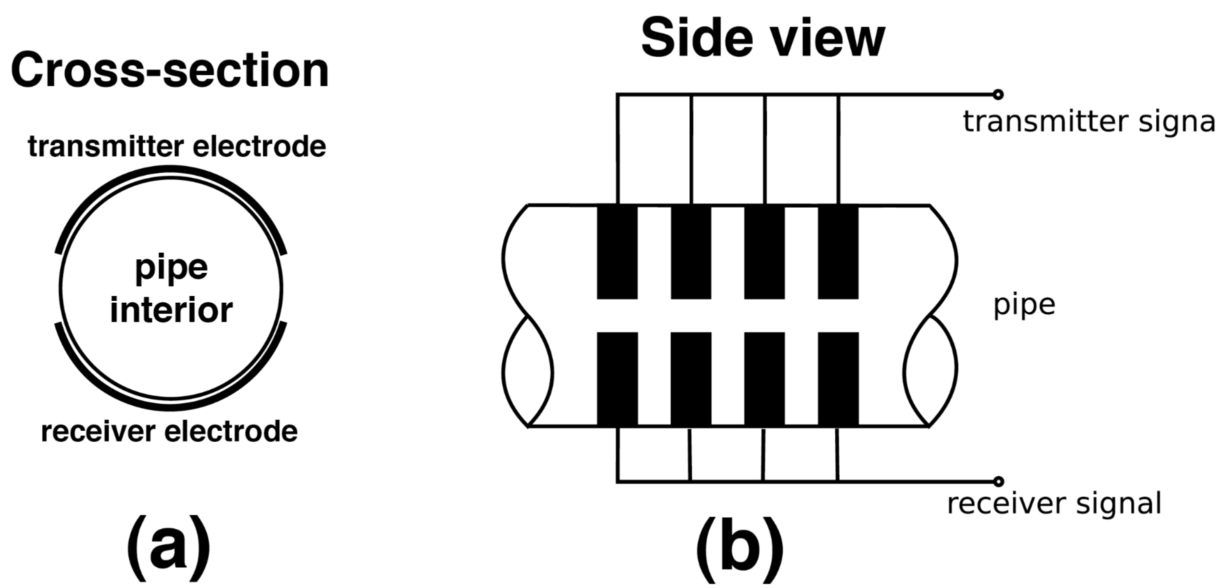


Fig. 2. Semi-cylindrical electrode capacitors, with differential excitation. Adapted from Hrach [1].

For our prototype, we will use a PVC pipe with a 3" inner diameter and 3.5" outer diameter. As grain flows down through the pipe past the capacitors, the changing relative permittivity ε_r changes the overall capacitance C of each capacitor.

Manaf and Triyana give a formula to calculate the capacitance of a semi-cylindrical electrode capacitor, Fig. 3(a).

Capacitance equation (for semi-cylindrical capacitor)

$$C = \frac{2 z \varepsilon_r \varepsilon_0}{\pi} \ln\left[\frac{\sin\theta}{(1-\cos\theta)}\right]$$

(a)

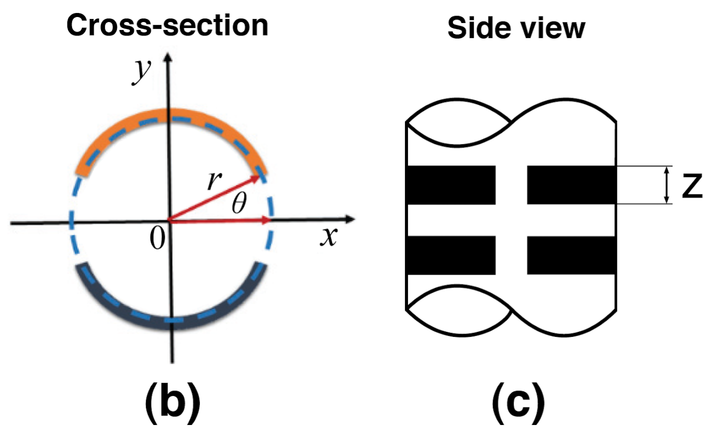


Fig. 3. Semi-cylindrical capacitor equation and geometry, adapted from Manaf and Triyana [2] and Hrach [1].

Oddly enough, this formula is independent of the radius r. Once an angle θ is chosen, as seen in Fig. 3(b), the size of the pipe radius is irrelevant to the capacitance. This is akin to taking the parallel-plate capacitor formula and fixing the width-to-distance ratio: C = A*ε/ d = (w * *l* * ε)/d, with area A = (width)*(length), and d = distance between plates. If you fix the ratio w/d to be constant, then the capacitance only depends on length *l*, no matter how close or how far apart the plates get, and no matter how wide each plate is..

To estimate the the range of capacitance, we assume the following:

- **Assumption 1**: A popcorn kernel has permittivity of $\varepsilon_r = 6.5$. Popcorn dielectric constant ε_r is around 4.75 to 8.25, depending on moisture content [3]. The higher the water content, the higher the dielectric. 6.5 is in the middle of that range.

- **Assumption 2**: Each capacitor has a vertical length of z = 3" = 0.0762 m. See Fig. 3(c).

- **Assumption 3**: The angle is θ = 10° = 0.174533 radians. In other words, plate 1 of each capacitor forms an arc on the circumference of the PVC pipe, from 10° to 170°, while plate 2 forms an arc from 190° to 350°.

- **Assumption 4**: We neglect the 0.5" curvature of PVC pipe inside the semi-cylindrical capacitor. The PVC, with a relative permittivity of 4.0, contributes to the capacitance [4]. However, the PVC thickness is set and doesn't change. Thus, the PVC arcs between the capacitor plates will not contribute to the variations of capacitance as grain flows through the pipe.

With these assumptions, our estimated capacitance is:

$C_{air} \approx 0.698$ pF

$C_{popcorn} \approx 6.8017$ pF

Difference $\Delta C = 6.10$ pF

From 100% air to 100% kernels, the capacitance would vary 6.10 pF. This is a high estimate, given that there will always be pockets of air between the kernels. Thus, the real change in capacitance will be lower than 6.10 pF.

We made out capacitors 3" long on the pipe, with about 3.25" distance between the four capacitors. We tested a single capacitor on our pipe (with nothing inside but air), and it had a capacitance of 100 pF. We then measured a change of $\Delta C = 4$ pF when inserting a bag of popcorn kernels. So the ΔC wasn't too far off the predictions. However, the capacitor in air value was drastically different: predicted was 0.698 pF, but measured was 100 pF. So perhaps Manaf and Triyana's non-peer-reviewed paper isn't accurate [2].

Amplifier

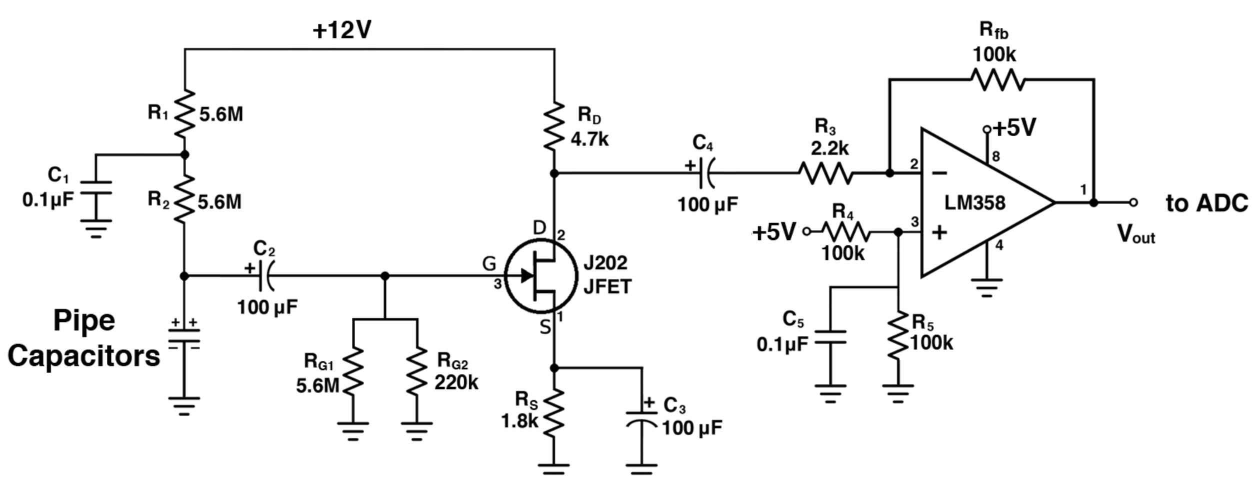


Fig. 4. Schematic of the amplifier circuit.

This module charges the capacitors and converts the changing capacitance into changing voltage. The pipe capacitors are charged with +12V.. DC-blocking capacitors ensure that only the changing AC signal (from grain flowing through the pipe) makes it to the JFET, and thus no direct current is going from the pipes to the JFET, and the full 12V DC voltage goes to the capacitors on the pipe.

Capacitor C1 was installed to ground, to prevent any voltage ripple from making it to the capacitor pipe. And the value of capacitor C2 and the resistor RG1 were made large enough so that the highpass filter going into the JFET had a low corner frequency below 1 Hz.

For simplicity, we used single-sided op-amps with 5V power, to match the 0 to 5V range of the Arduino's analog-to-digital converter.

Initially we thought that the op-amp would need a gain of at least 1000 V/V, in the range of 0 Hz to 1 kHz. This proved to be unnecessarily high amplification.

Via testing, we saw that our JFET gain was less than 2.
The inverting op-amp has a gain:

$$G_{op\text{-}amp} = - R_{fb} / R_3 = - 100k / 2.2k = 46 \text{ V/V}$$

The gains from the JFET and the op-amp are multiplied to get a total gain:

$$G_{JFET} * G_{op\text{-}amp} = 2*46 = 92$$

Our actual GAIN was 70 V/V, which is within the expected range.

The output stage of the amplifier must be a signal between 0V - 5V to accommodate the Arduino's analog-to-digital converter. Therefore, with a 5V single-sided op-amp, we will need a +2.5 DC-offset connected to + input, to shift a time-varying signal into the middle of our op-amp's headroom. We accomplished this with a voltage divider to take +5V to +2.5V.

Initially, our amplifier was built on a breadboard for design and testing. After successful testing, we planned to then transfer our design to PCB and order several boards from a fabrication company. All our components will be through-hole, with no surface mount parts.

Power Supply

We used a brick power supply connected to the 120V AC mains outlet. The power brick had both 12V and 5V DC outputs. This worked perfectly: 12V for the JFET and capacitor side of the circuit, and 5V for the op-amp half of the circuit.

The Arduino Uno was powered through the USB ports on our laptops for testing but could be connected to a wall outlet for implementation in the field. Since the cable is just a USB cable we would be able to find a connector block with the required voltage. The touch screen was also powered through the Arduino through the Arduino shield that we purchased with the touchscreen.

Memory

The Arduino Uno comes with 32 kilobytes (kB) of Flash memory. This will limit how big of a file we can save for the Flow Rate vs. Time graph. But the Arduino Uno's limited dynamic memory is incredibly small, at just 2 kB.

Analog-to-Digital Converter

Arduino Uno has a 10-bit ADC, in the voltage range of 0V - 5V. A 10-bit ADC has 1023 steps between 0 and 5 V, which allows for a sensitivity of 4.88 mV. Any voltage change less than 4.88 mV can't be read.

The native Arduino code for the ADC had a minimum 9kHz sample rate. However, we needed a much lower sample rate in the range of 150 Hz.

Touch Screen

We plan to use a 7" resistive touch screen of dimensions: 152 × 86 mm, 6.0" × 3.4", with a resolution: 800 x 480 pixels. We expect this resolution to be adequate for Ardent Mills staff to read the flow meter on-site, but if a higher resolution graph is needed, future updates to our device can include a larger screen or just send data to the mill control room, which will likely have full-sized computer monitors.

We chose a resistive touch screen rather than a capacitive touch screen, so that buttons can be pressed with gloved fingers.

The touchscreen "shield" is used to expand the capabilities of the Arduino. It was created to regulate the power to the screen and to protect both the screen and the Arduino if any strange power surges were to happen between the two.

Software Modules

Digital Signal Processing

For reading data in real time we used the Arduino function called millis() that returns the amount of milliseconds that have passed since program start, instead of the function delay() which pauses the program for an amount of milliseconds specified. With millis() we can ensure that the loop runs as often as we want, regardless of the execution time. With delay() this is not possible since we do not know how long the loop execution time is.

We implemented an if statement that checks when the desired sampling rate has elapsed; that is, if the time between the last ADC reading plus the READ_PERIOD (8 milliseconds) is less or equal to the current time.

```
if(millis() >= lastRead + READ_PERIOD){
        lastRead += READ_PERIOD;
        finalValue = analogRead(signalPin) - 485;
        fht_input[i] = finalValue * 64;
        i++;
        if (i == FHT_N)
        {
          Harley_Transform();
          Max_Value();
          i = 0;
          max_fht_lin = 0;_
          max_index = 0;
          max_freq = 0;
        }
    }
```

In this way, every 8 milliseconds we are reading values from the analog channel. The Arduino ADC creates a10-bit unsigned number: 0 to 1023, from which we then subtract the number 485 (2.37V offset), to create the equivalent of a signed 9-bit number, representing the signal with zero DC-offset.

The Arduino stores the ADC values as signed 15-bit numbers, despite the actual ADC range only covering a signed 9-bit number: -512 to +511. Therefore, before saving the ADC values to memory, we multiplied each one by 64 (which is 2^6), to increase the magnitude from +/- 2^9 to a full +/- 2^15 values: -32,768 to +3276.   Using the entire range of the signed 15-bit numbers helped with the DSP.

After the ADC, we need to take the signal from the time domain to the spatial frequency domain. We chose to use a Fast Hartley Transform (FHT), rather than the Fast Fourier Transform (FFT) because the FHT is specifically designed for real data, whereas the FFT operates on complex data. As a result, the FHT uses half as much processing power, and half as much memory. The FHT library allows a maximum of only 256 sample values, and only 128 frequency bins.

We chose a sample rate of f_s = 125 Hz. This allowed our ADC to capture frequencies from 0 Hz to 62.5 Hz. With the 128 frequency bins, with 62.5Hz maximum, each bin is 0.488 Hz wide. We feel that a bin size of around 0.5 Hz allowed us enough resolution to detect adequate changes in frequency in the lower 0 to 20 Hz band.

However, if 0.488 Hz bins proved to be too coarse, we could later lower the sample rate, which would decrease the size of the bins even further, allowing for finer distinction between the low frequencies.. Or if we used a more powerful non-Arduino processor in the future, we could engage a different FHT library with more sample points --- in the many thousand --- which would increase the frequency resolution.

Once the FHT is performed, we need to be certain to ignore the noisy signal and instead retrieve only the wanted signal. Our incoming signal from the amplifier circuit had a lot of 60 cycle hum (from the lights or power system.) After the processing has been completed using the FHT, we search for the maximum value of that output, and the index of that bin. The index of the maximum value gives us the frequency of the signal.

Once we get a frequency in hertz, we then will calibrate the meter. This will be done by measuring a known weight of popcorn through the pipe, while timing the flows in seconds. After knowing the flow rate in pounds/second, we can take the measured frequency from that specific trial. Together, these values will determine a mathematical constant to scale the frequency, to display a correct flow rate.

The final flow rate will be displayed on the GUI, but it can eventually be saved to the Arduino Flash memory, as an array or a .csv (comma separated value) file.

Graphical User Interface

The GUI on the touchscreen will have two main tabs:

- **Tab 1**. Flow Rate (instantaneous), which displays the current instantaneous flow rate, in large print.

- **Tab 2**. Graph (Flow Rate vs. Time). This graph will show the flow rate over time. We might include the option to select time intervals: the previous 15 minutes, 30 minutes, 1 hours, 3 hours, 12 hours. (Initially, however, during testing, we will probably graph 20 to 30 seconds, because we won't have a large enough dataset for longer periods.)

An important component of the GUI will be the on-screen warnings. Warnings will appear in whichever tab is currently being displayed on screen. There will be two warnings:

- **Warning 1**: The flow rate has gone outside specified bounds, (i.e. flow rate is too high, or too low.)

- **Warning 2**: The flow rate has stopped completely.

These warnings will be large on the screen, with each warning graphic probably blocking the regular information displayed. To clear a warning from the screen, a user must touch the warning block with a finger. The warning will then disappear.

The Arduino Uno has internal libraries that will help us create the GUI and the company that we purchased the touchscreen through also supplied libraries for us to use and examples for us to follow for the creation..

Validation and Acceptance Test Results

Circuit

For our circuit we successfully saw a change in voltage when different amounts of grain was poured through the pipe. This was done with an oscilloscope and testing different objects in the pipe to make sure the circuit could detect them going in and out of the pipe, this can be seen in Fig. 6. We were then able to validate that the circuit could amplify our signal with a gain of 70V/V. This was done by sending a waveform through our circuit with the Analog Discovery. Our output waveform can be seen in Fig. 8. Our circuit could also detect flow through the pipe. This successfully showed that our circuit worked properly and was ready to be hooked up to the arduino and the software.

Some issues we had with our circuit was that it was inconsistent. Our circuit ultimately did not work when hooked up to the Arduino. This was due to unknown causes because our circuit was not able to be seen or changed due to a group member covering it entirely in electrical tape. We believe there was an obscure wire or a part of our circuit that was not grounded correctly, or that the circuit wasn't attached to the Arduino correctly. For future design implementations we would have a PCB board instead of a breadboard. This would make sure that there were no loose wires and no blown parts, and our circuit would be seamless.

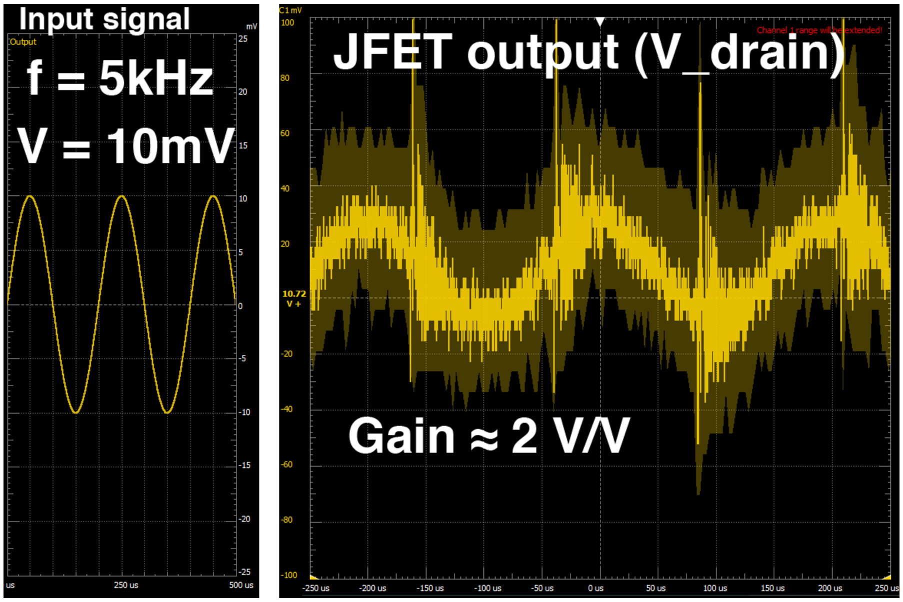


Fig. 5. JFET test. Input signal 10mV; output around 20 - 30 mV. Gain around 2 V/V.

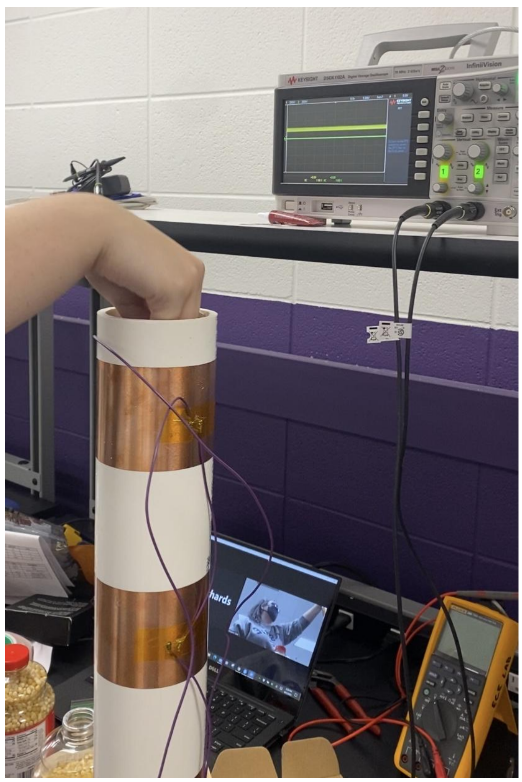


Fig. 6. Testing different objects in the pipe with the oscilloscope changing with the object.

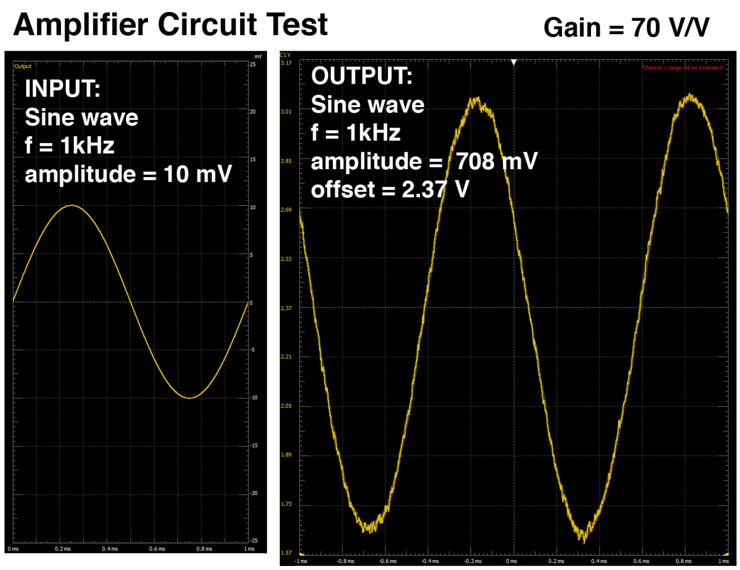


Fig. 7. Final amplifier test. Test signal input on JFET side, with the output on the op-amp side. Gain of 70 V/V.

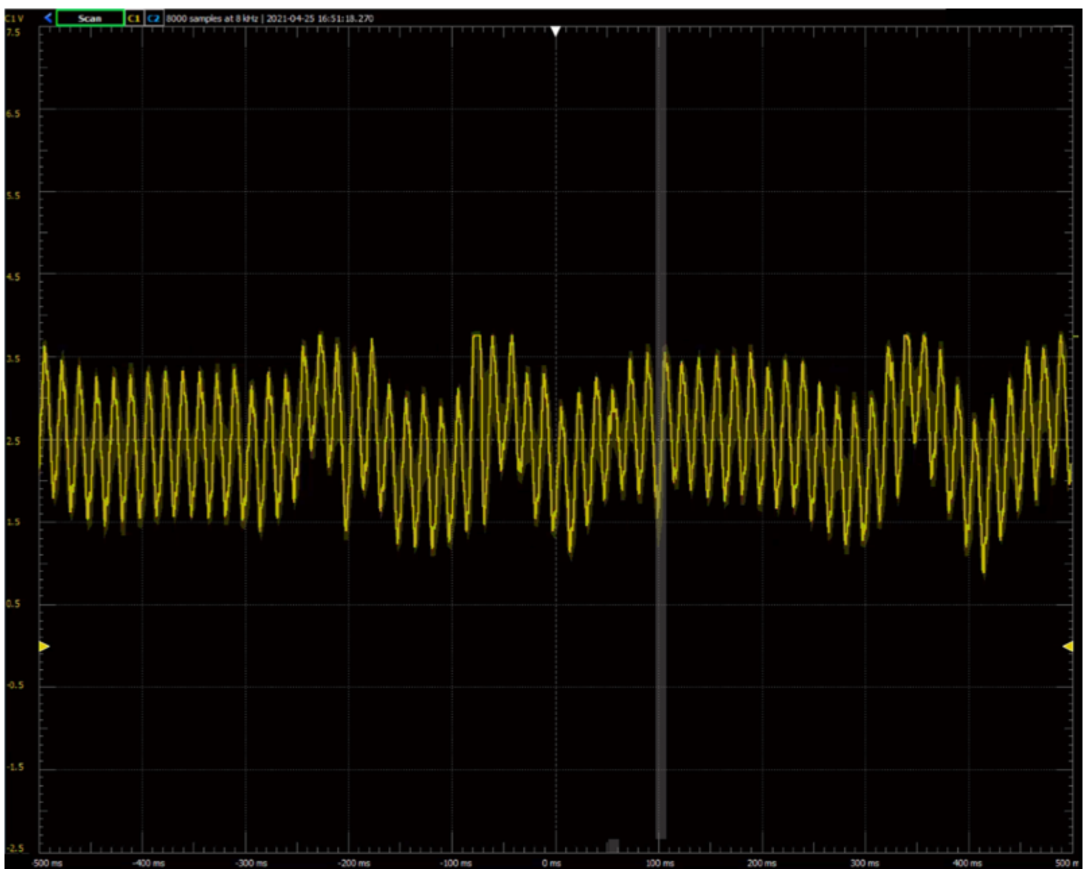


Fig. 8. Amplifier with capacitive pipe hooked up to it. The higher-frequency signal is 60-cycle hum from the power system. The time-changing capacitance (of a human hand rhythmically rubbing along the pipe) is seen as the lower frequency changes in voltage amplitude.

Software

Since we used an existing FHT Arduino library we first tested the functions of such to understand how to implement it for our purpose. Once we were able to do this, we used an Analog Discovery 2 to send signal waves into the Arduino ADC to validate the readings. In this way, we could then validate our digital signal processing by comparing the frequency we got as a result of the processing to the one we set in the analog discovery. We ran this testing a couple of times in order to choose our variables correctly for the calculations and reach the desired result, which was close enough.

The main issue we ran into was that the Arduino we were using did not have enough memory on it for our design. The TFT touchscreen used up more memory than expected. With this issue our software and our circuit were not able to be combined in the end. This being said, for future designs we would need an Arduino Mega to complete all that we wanted our flow meter to do.

```
21:18:19.138 -> Frequency = 45.41
21:18:19.138 -> good flow
21:18:21.185 -> Frequency = 44.43
21:18:21.185 -> good flow
21:18:23.224 -> Frequency = 43.46
21:18:23.224 -> good flow
21:18:25.265 -> Frequency = 42.48
21:18:25.265 -> good flow
21:18:27.306 -> Frequency = 1.46
21:18:27.306 -> too low
21:18:29.347 -> Frequency = 1.95
21:18:29.347 -> too low
21:18:31.392 -> Frequency = 2.93
21:18:31.439 -> too low
21:18:33.478 -> Frequency = 3.91
21:18:33.478 -> too low
21:18:35.518 -> Frequency = 4.88
21:18:35.518 -> too low
21:18:37.558 -> Frequency = 5.86
21:18:37.558 -> good flow
21:18:39.595 -> Frequency = 6.84
21:18:39.595 -> good flow
21:18:41.643 -> Frequency = 8.30
21:18:41.643 -> good flow
```

Fig. 9. The Arduino Uno DSP in action. Via the Arduino serial port output connected to a laptop, we displayed frequency readings, with warnings for low flow. (The signal coming into the Arduino ADC was a frequency sweep between 1Hz to 60Hz.)

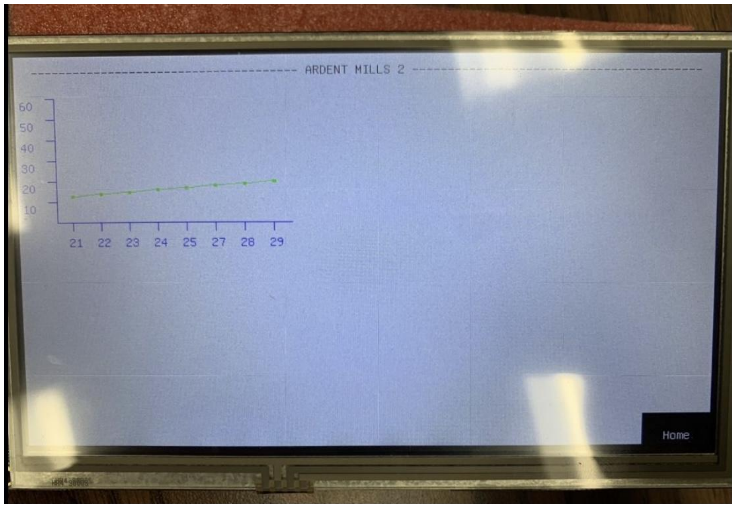


Fig. 10. Touchscreen display showing graphing of sample data every 2 seconds.

Fig. 11. Touchscreen display finished interface.

GLOBAL ISSUES

Ardent Mills has “mills, mixing facilities and a bakery, located throughout the United States, Canada and Puerto Rico” [5]. All three of these countries use the same 60Hz, 120V power system for homes. Our flow meter would not be employed in any countries that use different power plugs or jacks. Therefore, our device does not need any electrical modifications. If they were to deploy this device in a country or region that does have a different power plug it would be simple to change the power supply that we used to power the system to one that has a plug for the different region.

If Ardent Mills expands into regions with 50Hz power, then our DSP code would need to be changed to ignore 50-cycle hum, not 60-cycle hum and then be recalibrated to make sure the system continues to work the way intended.

Touch screen modifications

Summary: Our flow meter should be prepared to be deployed in any of Ardent Mills facilities around the world. Therefore, the GUI has to be available in different languages and metric systems.

Resolution: If our device is deployed in a different region than the United States, we could easily translate the GUI and on-screen warnings to the language needed in the country. Also, if the mill uses the metric system, we can easily convert the pounds / hour flow rate data into kilograms / hour.

WORK PLAN

Gantt Chart

Gantt Chart is located in the Appendix at the end of the document. In the chart each period is equal to one week.

Work Breakdown Structure

Task #	Task Description	Meghan	Luis	Valeria	Jenalee	Joshua
1.0	**Hardware Validation**					
1.1	Capacitance Sensor	X			X	X
1.2	Touchscreen Monitor	X	X		X	
2	**Arduino Software**					
2.1	Sensor Processing (DSP)			X		X
2.2	Touchscreen Software Setup		X	X		
2.3	GUI Implementation		X	X		
2.4	Warning Detection Setup		X	X		

3	**Physical Design**					
3.1	Breadboard/PCB design	**X**			**X**	**X**
3.2	Circuit Design	**X**			**X**	**X**
4	**Software Validation**					
4.1	Touch Screen works properly		**X**	**X**		
4.2	Data gets recorded		**X**	**X**		**X**
4.3	Graph (Flow vs Time) output		**X**	**X**		

Financial Plan

Item #	**Item/ Part Description**	**Cost**	**Supplier**
1	Arduino Uno	0.00	Supplied by Department
2	Touchscreen Display	82.28	BuyDisplay
3	3” x 24” PVC Pipe	5.60	HomeDepot
4	Popcorn Grains for testing (6 jars)	25.68	Amazon
5	Solderless Breadboard	0.00	Supplied by Department
7	Wiring	0.00	Supplied by Department
8	JFET: J202 (2) through-hole	0.00	Supplied by Group Member
9	Copper Tape for Capacitors	11.49	Amazon
10	Kapton / Insulation Tape	10.00	Amazon
11	12V / 5V Power Supply	0.00	Supplied by Group Member
12	OpAmp: LM358 (3) PDIP through-hole	2.76	DigiKey/Mouser
13	Resistors/ Capacitors	0.00	Supplied by Department
Total		127.81	+shipping costs

Originally we believed our budget to be $251.42, but we were able to get our project finished for $127.81 plus shipping thanks to the ECE department supplying many of the smaller parts out of parts they already had. These parts supplied by the department are highlighted in the table above. We believe that we could produce multiple flow meters with these supplies.

Feasibility Assessment (resources and risks)

During our assessment of the project's strengths and weaknesses we found a few to discuss. One of the huge strengths the project has is that there are no moving parts. This increases the lifespan of the device so it will not have to be replaced as often as other flow meters that are more readily available to this industry. Another strength is how fast and efficient our design will be because we won't have to stop and wait for the grain to be weighed and dumped into the next hopper, it will be a constant flow of grains instead.

One of the risks was the complex digital signal processing to pull the capacitive signal from the noise. However, we accomplished the frequency detection DSP while ignoring the 60-cycle noise. Another risk was that testing and troubleshooting might be difficult because not every group member can meet in person because of health issues related to the COVID-19 pandemic. This risk was born out.

Another risk that we might encounter is our sensor is unlikely to be cost efficient due to the existing commercial products. Looking at the strengths and weaknesses of this project, there are multiple solutions at our disposal to help us create our project, so this makes our project feasible.

Lessons Learned & Recommendations

1. The Arduino Uno was not an adequate processor for our project.

The Arduino Uno only has 2 kilobytes of dynamic memory for variables. This made it impossible for the DSP/FHT code and the touchscreen code to run at the same time. We should have used an Arduino Mega 2560, or another processor altogether.

For example, due to the Arduino Uno's limited memory, the Fast Hartley Transform had a maximum of 256 sample points, and only 128 unique frequency bins. This limited the sample rate to be much lower than expected. We were able to work around this limitation, but having more FHT sample points would have allowed us more freedom in the DSP

2. The touchscreen was too big.

We could have used a smaller screen. The large touchscreen took up too many of the Arduino Uno ports. So either we needed a smaller touchscreen or a bigger Arduino.

3. Better op-amp

We chose an older design of op-amp which was not "rail-to-rail." Our LM358 op-amp was powered by 5V, but its output level was limited to around 3.8V to 4V --- a full 1 V less than the power supply. Had we chosen a more modern rail-to-rail op-amp, we could have amplified our signal closer to +5V without clipping.

4. COVID issues

The circuit designer was remote, due to COVID. We found that trouble-shooting circuit issues over Zoom video was extraordinarily difficult. An ideal situation is one where all team members can meet in person.

5. Poor time management

When certain team members didn't turn in work on time, it created a bottleneck for other students, and the workload kept getting pushed back later and later into the semester. By the time that the circuit was completely built, there wasn't enough time for trouble-shooting and integration of the project as a whole.

The same goes for the touchscreen implementation: we didn't get the on-screen warnings to function because we ran out of time. In the future, if we were to continue with the sensor project, we would add the on-screen warnings.

Conclusion

The goal of our project was to create a cheaper flow meter that Ardent Mills can implement into their mills. Our system was designed to read the flow through the meter and output it to our touchscreen display. The system could keep track of the flow rate over different periods of time. We weren't able to fully test this part of the system due to time constraints due to being students and also the limit of the space on our Arduino. We believe that it should work since we were able to test these ideas separately. The data is available on the touchscreen by selecting the time period you want to view. There are ways that the Arduino could connect to the Ardent Mills factory control system eventually but since we are unsure of what they use for that we were unable to implement any of that during this build.

REFERENCES

[1] D. Hrach, A. Fuchs and H. Zangl, "Capacitive flowmeter for gas-solids flow applications exploiting spatial frequency," *2008 IEEE Sensors Applications Symposium*, Atlanta, GA, 2008, pp. 26-30, doi: 10.1109/SAS13374.2008.4472937.

[2] M. N. Manaf and K. Triyana, "Analytical solutions for capacitance of a semi-cylindrical capacitive sensor," *AIP Conference Proceedings*, 2016, vol. 1755, no. 1, pp. 020002-1 – 020002-5, doi: 10.1063/1.4958467.

[3] J. Novák, "Electrical properties of popcorn grains," *Acta Technologica Agriculturae*, 2013 vol. 16, no. 2, pp. 26-30, doi: 10.2478/ata-2013-0011.

[4] "What is a Dielectric Constant of Plastic Materials?" https://passive-components.eu/what-is-dielectric-constant-of-plastic-materials , (accessed December 2, 2020.)

[5] "Ardent Mills: Our Facilities," https://www.ardentmills.com/our-facilities, (accessed on December 3, 2020.)

[6] "Ardent Mills | Flour Milling & Grain Innovations." http://www.ardentmills.com/ (accessed Oct. 17, 2020).

[7] Y. Yan, "Mass flow measurement of bulk solids in pneumatic pipelines," *Meas. Sci. Technol.*, 1996, vol. 7, no. 12, pp. 1687–1706, doi: 10.1088/0957-0233/7/12/002.

[8] Zhang, "Air-Solids Flow Measurement Using Electrostatic Techniques," in *Electrostatics*, H. Canbolat (Ed.), InTech: 2012, pp. 61-80, doi: 10.5772/35937.

[9] W.Y. Du, *Resistive, Capacitive, Inductive, and Magnetic Sensor Technologies*, 1st ed. Boca Raton, FL, USA: CRC Press, 2015.

[10] https://web.archive.org/web/20171203144114/http://www.usb.org/developers/docs/usb20_docs/

APPENDIX

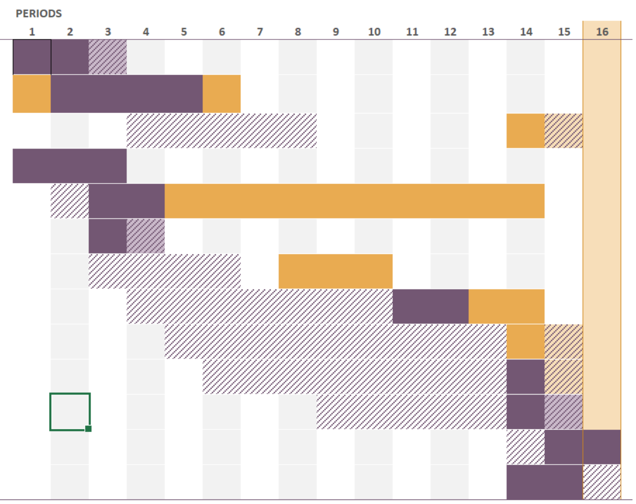
Team 4- Project Planner

Select a period to highlight at right. A legend describing the charting follows.

Period Highlight: 16

Plan Duration | Actual Start | % Complete | Actual (beyond plan) | % Complete (beyond plan)

ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Requirements	1	3	1	3	95%
Design of Capacitive Flow Meter	2	4	1	6	100%
Digital Signal Processing	4	5	14	2	70%
Cost/Budget Prep	1	3	1	3	100%
Arduino Software Development	2	3	3	12	100%
Touchscreen Set Up	3	2	3	2	95%
Circuit Set Up	3	4	8	3	100%
Electronics Implementation	4	9	11	4	100%
Software Implementation	5	9	14	2	90%
System Integration	6	9	14	2	50%
Testing & Verification	9	7	14	2	50%
Final Documentation	14	3	15	2	100%
Final Presentation	14	3	14	2	100%



DSP Code

https://github.com/vmorinigo/SeniorDesignProject_FlowMeter.git