Data Acquisition System For Soil Sensors

Final Report

Team #5

Client
Dr. Halil Ceylan

Advisers
Shuo Yang & Dr. Yang Zhang

Team Members
Nathan Coonrod (Report Manager and Hardware Lead)
Kyle Kehoe (Communications Manager and Testing Lead)
Jacob Verheyen (Meeting Facilitator and Hardware Designer)
David Severson (Web Master and Reliability Lead)
Sok Yan Poon (Timeline Manager)

Team Email
sddec18-05@iastate.edu

Team Website
sddec18-05.sd.ece.iastate.edu

Table of Contents

List of Figures

List of Tables

1. Introductory Material

1.1. Problem Statement

The Civil, Construction, and Environmental Engineering (CCEE) Department is currently using expensive and unreliable sensors for monitoring soil moisture and temperature which do not come with data acquisition systems. Our project is to design a reliable, inexpensive, and weatherproof data acquisition (DAQ) system for underground soil temperature and moisture monitoring. Our team was originally using MEMS sensors that were previously developed by the ECpE department. Unfortunately, the MEMS sensors were not durable enough for the application. Therefore, our team also delivered an alternative sensor to interface with our DAQ design which was outside the scope of the project.

1.2. Operating Environment

The sensor and data acquisition module will be used outside in the harsh Iowa environment. The DAQ will be located above ground and will be connected to the underground sensors via cables. The DAQ, SD card and battery will be housed in a weatherproof container.

1.3 Market Research and Commercial Alternatives

The data acquisition system was created to be a “set it and forget it” type of system. Our client emphasized how expensive DAQ systems are, and those that continuously read real-time data require a researcher to be outside logging points at various intervals. They would like to be able to set the system in the field for durations of up to a month and simply retrieve an SD card to observe the soil characteristics a month later with confidence that the data they want was recorded.

Our customer has used commercially available DAQs in the past. However, none of the tested products have been viable for the customer’s application due to various issues. Below are two of the several systems our client has used:

ECH2O Datalogger: $750/Module, $206/Sensor
- 5 channels
- 1-3 year battery life depending on sample rate
- +/- 1 degrees Celsius
- +/- 3% moisture content

Sensirion EK-H4: $230/Module, $159/Sensor
- 4 channels of temp and humidity sensors
- +/- 3% relative humidity
- +/- 0.4 degrees Celsius
- Requires AC outlet

Our product takes the best qualities from both of these products to create a specialized product for our client. The design is able to log data over a long period of time at a flexible interval while being inexpensive, reliable, and scalable for large projects requiring many DAQs.

2. Requirements

2.1. Functional

1. 2 channels of temperature measurement
2. 2 channels of moisture measurement
3. Rechargeable internal battery
4. Detachable sensors for remote placement
5. Waterproof enclosure and sensors
6. Battery charging indicator
7. DAQ status LED
8. Onboard Real Time Clock (RTC) for record date stamps

2.2. Non-Functional

1. Sample period of 15 minutes
2. Average battery life of 1 month
3. Analog sensor interface
4. Minimum cable length of 6”

3. Project Design

3.1. DAQ Design

After iterating through several design prototypes, the team settled on a final architecture which would yield a reliable, easy to manufacture, and low-cost solution. The core of the DAQ, the ATMEGA32u4 microcontroller, remained consistent throughout all prototypes which reduced software changes between designs. All design decisions were made with our customers' needs in mind. The design is intended to be a simple solution which avoids the pitfalls of commercial off-the-shelf (COTS) solutions while remaining easy to modify in software for future applications and challenges faced by the CCEE department.

3.2. Soil Sensor Design

The group had several design prototypes for our sensors before settling on a final design. Our temperature measurement circuit remained consistent, but we tried several different circuits for measuring moisture content. Our temperature measurement circuit uses a thermistor to measure temperature. To measure moisture, we measure the capacitance of the soil and convert that capacitance to a voltage.

3.3. Software Design

In order to keep the project software simple and easy to iterate in the future, an Arduino based design was chosen. This has the added benefit of a large array of available development boards which made prototyping software less complex. We had the ability to start software development in the first few weeks of the project, long before any hardware prototypes existed. That decision also decreased our risk when designing hardware prototypes. If we had chosen to use a more specialized microcontroller, we likely would never have had the ability to test software before the hardware existed. That would have introduced additional risk of hardware issues. Additionally, our customer may need to update the software on the DAQ to work in a specific environment. The Arduino IDE is the most user-friendly way our customer can change these parameters and settings.

4. Implementation Details

4.1. DAQ Implementation

The final architecture uses an energy dense lithium ion cell paired with a real-time clock, Atmel microcontroller, and the following peripherals:

- Micro SD card
- Micro USB for charging and programming
- Battery charging and protection circuits
- Analog signal conditioning for all sensor channels
- Real time clock

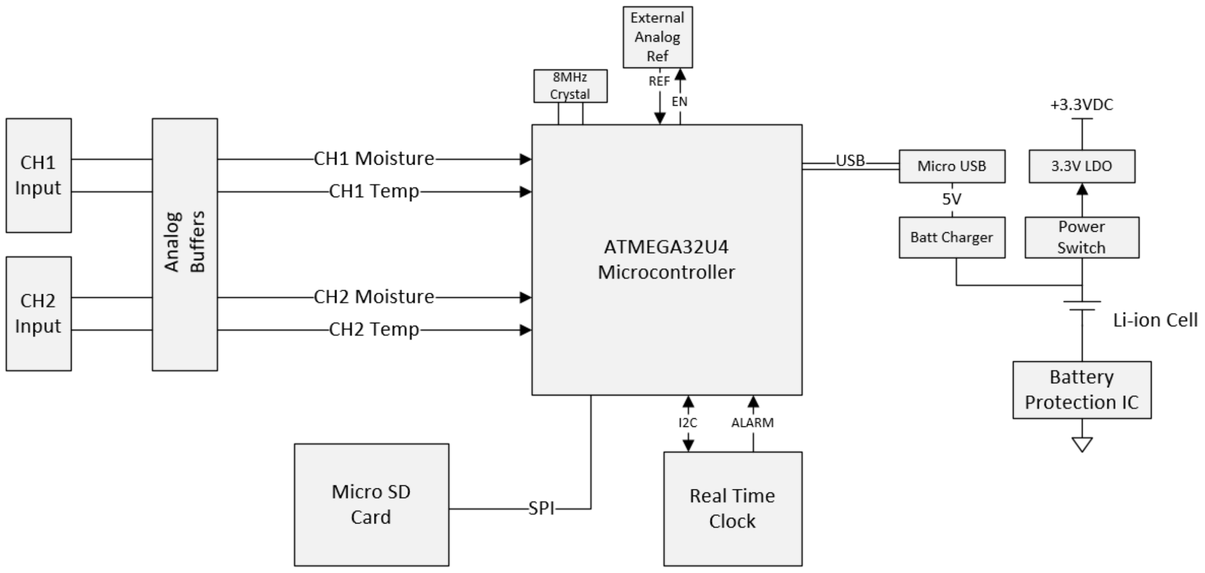


Figure 1: DAQ Architecture

A low power solution is achieved using the combination of microcontroller, lithium ion cell, and real time clock (RTC). The RTC has a feature which allows the microcontroller to set an 'alarm' which when triggered, creates a falling edge on a line which the microcontroller can use to wake from deep sleep. This allows the microcontroller to be in a very low power sleep mode for most of its life, waking whenever the alarm is triggered. Once the microcontroller is awake, it takes all needed measurements, records relevant data to the SD card, sets the alarm to trigger in 15 minutes, and puts itself back to sleep.

The battery life is dependent upon several characteristics of the DAQ. Run time can be derived by dividing the battery capacity by the average power draw. The battery is removable from the DAQ, so any 18650 size lithium ion cell is compatible. An ICR18650A cell was delivered to our customer in the final design. This specific cell has a nominal capacity of 2.6 Ah. In order to use the entire battery capacity, the DAQ requires a switching power supply because the 3.3V rail is in the middle of the lithium ion cell upper and lower voltage limits. The tradeoff of using a switching supply, is a higher quiescent current. Since our design is targeting an ultra-low average current, the benefits of a low quiescent current linear regulator (LDO) offset the benefits of using the entire battery capacity. We found that most lithium ion cells have under 10% of their capacity beneath the LDO's dropout voltage. This design decision allowed us to choose an LDO with a typical quiescent current of 2 uA as opposed to a typical quiescent current of over 20 uA for similarly rated switching power supplies. LDO regulators are also much less expensive than switching supplies due to the switching supplies' increased complexity and need for large ceramic capacitors and inductors.

Switching supplies have the advantage of being much more efficient under load, however, the current draw of the DAQ when in operation is moderately low, well under 100mA. Also, due to the low duration of on time vs off time, the power supply efficiency did not critically affect the power supply choice. Our team is confident that our final design is the best balance of battery performance, cost, and reliability.

The design will spend nearly its entire life in a low power sleep state. The hardware was designed to be able to shut off many of the peripherals when in this mode to limit power draw. The hardware is able to completely power down the sensors connected to the front panel as well as the microcontrollers own analog reference. All other power savings are obtained by putting the microcontroller in the lowest power sleep mode available. Doing so cut our current draw from ~25 mA for the whole system to ~1.6 mA, nearly a 94% savings. The remaining power is from the real time clock, battery management system, regulator quiescent current, microcontroller sleep current, and analog front end quiescent current.

The DAQ is required to be waterproof due to the environment it will be used in. There are numerous ways to environmentally isolate electronics, we chose a method which is low-cost and requires minimal manufacturing or assembly work. With most enclosures, holes would need to be drilled in the front panel manually, labeling applied to the exterior, the PCB mounted to the interior of the enclosure via standoffs, and an internal wire harness created to connect the panel connectors/indicators to the PCB. Our solution eliminates most of that by using the PCB as the front panel. This means that connectors, switches, indicators, and labeling can be directly placed on the PCB eliminating the internal wiring harness, PCB mounts, and many manufacturing steps. The resulting product has a very professional look and is much easier to manufacture and assemble which will reduce failures due to poor manufacturing on future DAQs.

Using a PCB as a front panel is not without challenges, but in this case we found that the challenges were far outweighed by the advantages listed above. Some challenges we had to overcome to use this method include:

- Waterproof gasket between PCB and enclosure
- PCB can have no open through-holes which conduct water
    - Vias were made minimum size manufacturer supported and covered with soldermask to "plug" them
    - All through-holes required by component footprints are filled with solder
- Indication LEDs are not robust enough to be placed on surface

In order to preserve the ability to use LEDs to indicate DAQ status, we used reverse mounted LEDs which point towards the PCB they're mounted on and removed all soldermask and copper on layers beneath them. This allows the LEDs to shine through the board and be clearly visible on the other side while remaining completely waterproof.

To keep costs low, the DAQ utilizes a standardized connector for each channel which allows low-cost off-the-shelf waterproof cables to be used. M12 connectors were chosen for their small size, relatively low-cost, waterproof housings, and widely available cables. M12 connectors are standardized across many manufacturers and have an easy to understand keying system which guarantees the connectors can only mate to sensor cables in one orientation. The power switch also needed to be waterproof, so a potted and waterproof switch was chosen.

Once the DAQ is recovered from the field, the user will likely want to recharge the internal battery as well as remove the SD card. Due to the requirement for low-cost, it was decided that rather than use interfaces which are waterproof, they use standard, non-waterproof, micro USB and SD card connectors on the inside of the DAQ. Consequently, the user must remove the front panel in order to access either. In order to keep the DAQ easy to use, the front panel was designed such that it requires no tools to open. The panel is held against the enclosure and gasket using four thumb screws, one on each corner of the enclosure front panel.

4.2. Sensor Implementation

The soil sensor was designed to easily integrate with the DAQ system, both electrically and mechanically. In order to mechanically connect a sensor to the DAQ, we chose to use six-foot-long weather-proof cables that can withstand the rough Iowa weather. Each cable has a 3.3V power wire, a ground wire, a wire to transmit moisture content, and a wire to transmit temperature.

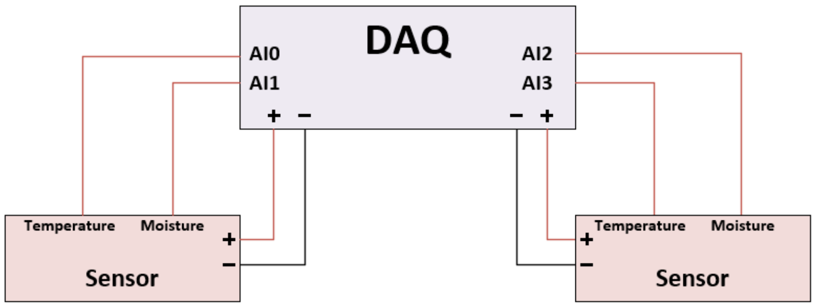


Figure 2: System Block Diagram

The 3.3V line will be supplied by the DAQ. Our sensor will output a voltage between 0 and 3.3V that corresponds to the temperature, and a voltage between 0 and 3.3V that corresponds to the moisture content. The main reason we chose to use analog voltage outputs rather than a digital communication protocol to transmit temperature and moisture content to the DAQ was design simplicity. We knew that it would be easy to generate an analog voltage that could be read by the ADC on our DAQ; we did not know what problems we might have encountered by trying to add a digital transmitter. We also knew that the ADC on the DAQ has virtually zero input bias current, so the voltage drop across our cables would be negligible.

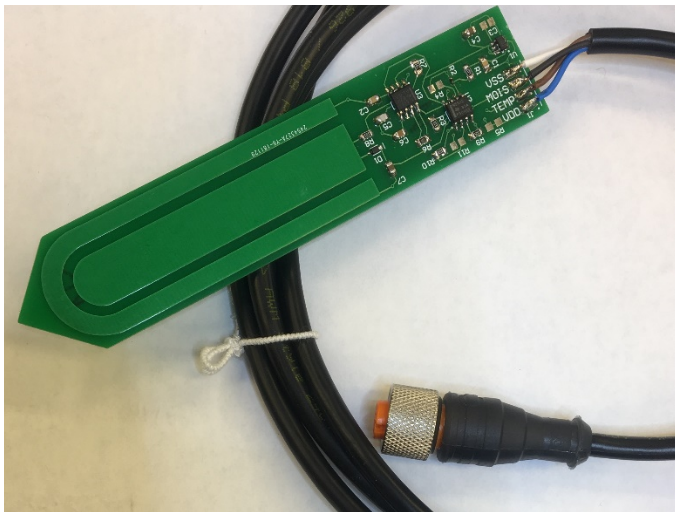


Figure 3: Sensor with cable attached

The temperature measurement circuit on our sensor is based on a negative temperature coefficient (NTC) thermistor. We chose to use a thermistor instead of a thermocouple or other alternatives because it was cheap and easy to implement. It also still gives us good resolution. We put a buffer amplifier at the output of our sensor to ensure any bias current flowing through our cables would not impact our temperature measurements.

In order to measure the moisture content, we relied on the fact that the capacitance of soil increases as its moisture content increases. We fabricated a sensor based on an open-source design that we found online; it has probes that extend out of it into the soil, and we measure the capacitance between those two probes. The circuit uses a 555 timer to generate a square wave which is then fed into an RC low-pass filter, where the capacitor in that filter is actually the capacitance of the soil. The output of that filter is then fed into a peak detector, then a buffer. As the moisture content increases, so will the capacitance. As capacitance increases, the output of the RC filter and the output of the peak detector will decrease. This means that as the moisture content of the soil increases, we will see a decrease in the output voltage.

4.3. Software Implementation

Our software was implemented in a way to meet customer requirements for our DAQ including sample period of data measurements and battery life of the system. Specifically, we were tasked with taking a sample every 15 minutes and for each sample recording 2 temperature readings in the soil and 2 moisture readings along with recording the time the sample was taken. The battery life of our system was required to last at least one month in good weather conditions.

To meet the requirements, an Arduino datalogging example sketch (program) was modified to write comma separated temperature and moisture values to a new line of a .txt file stored on an SD card once the ADC voltage readings were adjusted accordingly. This took care of storing the data and recording it, but we still need to address how the data gets timestamped every 15 minutes and conserve power, so the system can meet the 1-month battery requirement. An external peripheral DS3231 Real Time Clock (RTC) was used as a timekeeping device and as an interrupt signal generator since it has a programmable alarm output line that is normally “HIGH” but can be programmed to go “LOW’ when the correct time conditions are met. With the programmable alarm feature of the RTC, we now had a way to wake up our microcontroller from a deep sleep power saving mode every 15 minutes to record a new data value, in other words, the microcontroller only needs to be on every 15th minute to record a new data value and then can be put to sleep for another 15 minutes.

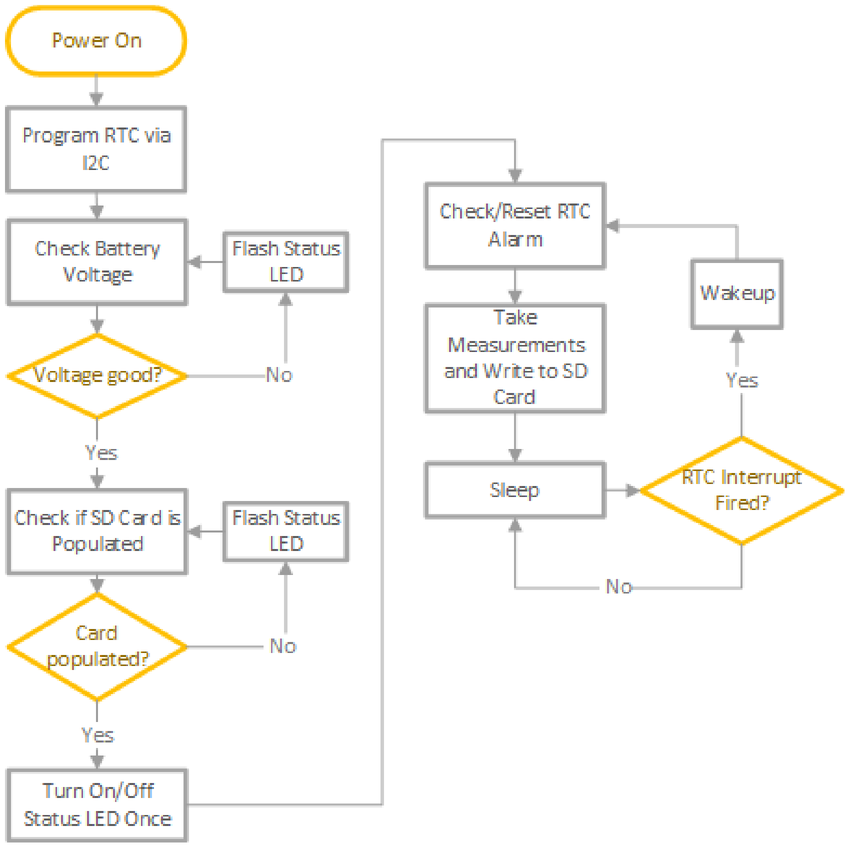


Figure 4: Software State Diagram

5. Testing Process and Results

DAQ Testing Results

Table 1 (shown below) shows different power draws for three different DAQ conditions. The first condition is when the DAQ is turned off; DAQ is in sleep mode where power switch is on; and the condition when current drawn for short amount of time while microcontroller wakes up to take readings before going back to sleep mode. All static power consumption measurements were taken with a 6.5 digit DMM using a shunt resistor of 10 ohms in series with the DAQ battery. For dynamic power consumption, oscilloscope probes were connected to either side of the shunt resistor and subtracted using an oscilloscope math function. This allowed us to view the real time consumption as the DAQ exits sleep mode, communicates with various peripherals, and goes back to sleep. We were also able to use the measurement period at just under 12ms. Results of this testing can be seen in *Table 1* as well as *Figure 5*.

R(ohm)	V(mV)	I(µA)	Notes
10	0.052	5.2	*Condition 1:* DAQ is turned off and current value is 5.2µA. Battery protection in circuitry and RTC are sole loads.
10	16	1,600	*Condition 2:* DAQ is in sleep mode. The battery draw is 1.6 mA.
10	290	29,000	*Condition 3:* Differential probe technique. To ascertain spike when it turns on to take data. Averaging approximately 29 mA. Output shown in *Figure 5*.

Table 1: Power draw testing results

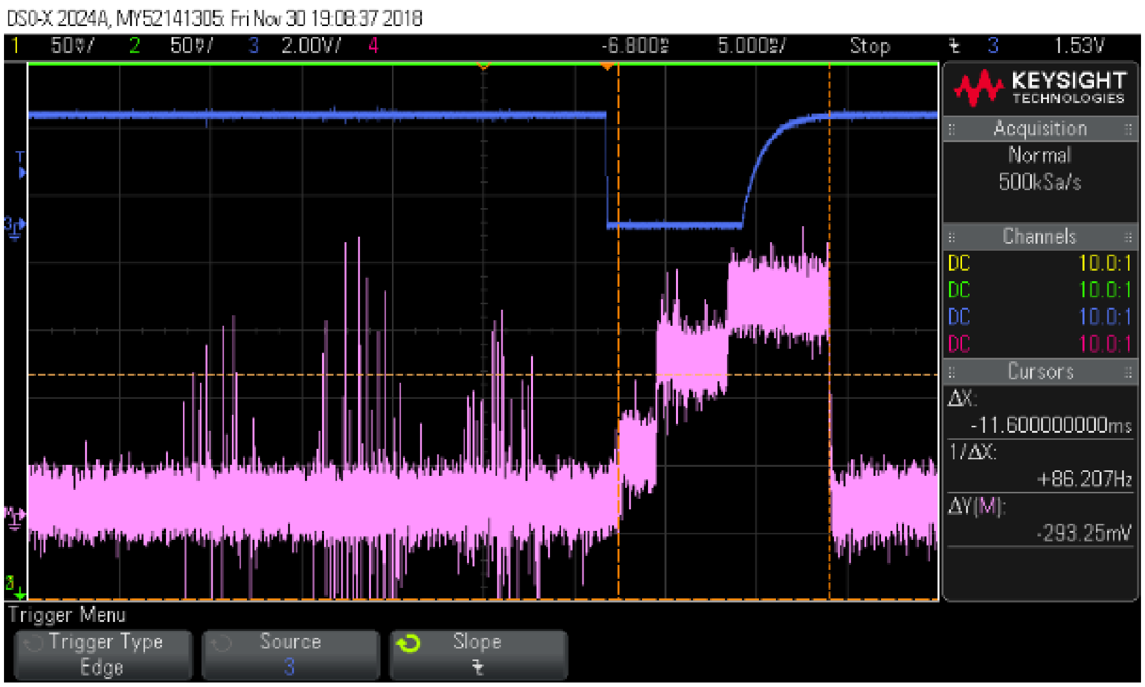


Figure 5: DAQ dynamic power consumption

The battery life requirement can be verified using the current consumption data collected above. Calculation for battery life in days in ideal conditions calculated with the following equation:

$$Battery\ Life\ (days) = \frac{2.6Ah}{\frac{1.6mA}{24hours}} = 67.7\ days$$

Calculation for battery life in cold weather with estimated battery capacity correction factor 0.5:

$$Battery\ Life\ (days) = \frac{2.6Ah}{\frac{1.6mA}{24hours}} * 0.5 = 33.85\ days$$

In both situations the requirement of one-month battery life is exceeded. Should the user desire more battery life, a higher capacity battery can be used. Lithium Ion cells are available up to approximately 3.4Ah which would give an estimated lifetime of 88 days in fair conditions.

The DAQ relies on a Low-Dropout (LDO) regulator to generate the internal 3.3V rail used by the microcontroller and all peripherals. Because the battery voltage is so close to this internal rail, we were careful to verify that our design would still operate. The lithium ion cell used has a voltage range of 2.8 to 4.2V, with a rail voltage of 3.3V, it's important to determine at what point the LDO can no longer regulate. Using our DAQ and a resistor across the LDO output to simulate a large static load, we decided that we would consider the rail as 'out of regulation' if it was more than +/- 1% from the nominal 3.3V. The datasheet only specified the dropout voltage at the full load current, far higher than any expected draw in our application.

Using a load of ~25mA, we measured the dropout voltage to be only 80mV. That is, the 3.3V rail fell out of regulation if the battery voltage was below 3.38V. This test verified that our design can use nearly all of the battery capacity before falling out of regulation. In order to prevent operation with a low battery, the DAQ software measures the battery voltage before every sensor measurement and will cease measurement if the battery falls below 3.4V. As further protection, the hardware battery protection circuit will cut off any load if the cell falls below 2.8V.

Testing Results for Temperature Sensors

Temp Setpoint (F)	Chamber Measured Temp (F)	Thermometer Measured Temp (F)	Output Voltage	Measured temp (C)	Measured temp (F)	Error 1 (F)	Error 2 (F)
-30	-30.1	-25.3	2.3157	-33.28	-27.9	-2.2	2.6
-20	-20.2	-14.1	2.2360	-27.83	-18.09	-2.11	3.99
-10	-10.2	-4.99	2.1482	-23.1	-9.58	-0.62	4.59
0	-0.1	5.7	2.026	-17.76	0.032	-0.132	5.668
10	9.7	15.2	1.898	-13.01	8.58	1.12	6.62
20	19.8	25.3	1.745	-7.97	17.65	2.15	7.65
30	29.5	33.74	1.576	-2.853	26.856	2.644	6.884
40	39.4	43.9	1.4	2.269	36.084	3.316	7.816
50	49.2	54	1.216	7.64	45.75	3.45	8.25
60	59.6	64.2	1.031	13.3	55.94	3.66	8.26
70	69.7	74.3	0.866	18.83	65.89	3.81	8.41
80	79.8	83.9	0.724	24.18	75.52	4.28	8.38
90	89.8	93.8	0.6001	29.58	85.24	4.56	8.56
100	99.6	104.1	0.498	34.82	94.68	4.92	9.42

Table 2: Temperature characterization data

Error 1 = Chamber Measured Temperature (F) – DAQ Measured Temperature (F)
Error 2 = Thermometer Measured Temp (F) – DAQ Measured Temperature (F)

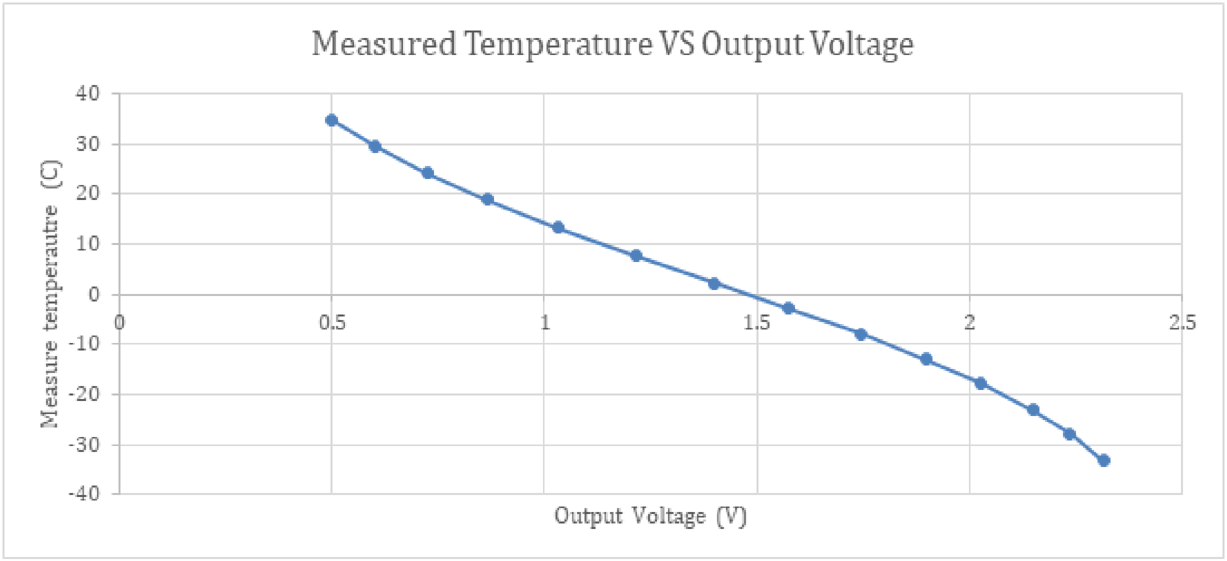


Figure 6: Measured temperature vs output voltage

The below equation was used to determine the ideal output voltage of the temperature measurement circuit by replacing the variable x with the current temperature in degrees Celsius. The sensor response was further validated using a PSpice model as well as the actual sensors.

$$\frac{2.5\times 100k\times e^{\left(4250\left(\frac{1}{x+273}\right)-\left(\frac{1}{293}\right)\right)}}{200k+100k\times\left(e^{\left(4250\left(\frac{1}{x+273}\right)-\left(\frac{1}{293}\right)\right)}\right)}$$

Moisture Sensor Testing Results

Characterization of the moisture sensor is dependent on the medium in which the sensor will be used. The general testing procedure for moisture sensor characterization is to take a completely dry soil sample that we know the volume or mass of in a controlled environment and get a reading from our moisture sensor and iteratively apply a known amount of moisture and record subsequent voltage readings from our sensor. Iterating this procedure, we can get enough data points to create a trend line that can be used in future software development to output moisture content levels. The researchers will need to characterize one sensor in each medium prior to deploying additional sensors.

Additional Tests Performed

Temperature Independence Test Procedure:

1. Place DAQ inside temperature chamber capable of minimum 0 to 100F

2. Connect reference resistor and capacitor to DAQ input terminals. Resistor should be approximately 250 ohms, capacitor should be approximately 1pF Note: reference resistor and capacitor should be outside temperature chamber

3. Configure DAQ to continually output measurements to PC serial monitor every one second

4. Set temperature chamber to sweep over 0 to 100 degrees Fahrenheit.

5. Monitor serial monitor and record any change in measured value and the temperature at which the change occurred

Success Criteria: Output for both temperature and moisture changed by no more than 1 LSB over the entire temperature range

Failure Criteria: Output for either reference component changed by more than 1 LSB at any point in the test

Sample Period Test Procedure:

1. In a lab environment, configure UUT for 15-minute sample period

2. Configure UUT for 4 samples

3. Attach two temperature and two moisture sensors Note: test does not depend on recorded sensor data
4. Start datalogger

5. One hour later, retrieve and open SD card on PC

Success Criteria: Four samples exist for each channel and samples occur every 15 minutes with 5 second margin of error

Failure Criteria: Any number of samples less than or greater than four or recorded time stamp is not in 15 minute increments

Storage Test Procedure:

1. In a lab environment, attach two temperature and two moisture sensors to the UUT (unit under test) Note: test does not depend on sensor data

2. Configure the UUT for a sample period of 1 second by editing the CONFIG.txt file on the SD card

3. Configure the UUT for a sample length of 3000 samples. Note: 3000 derived from 15 minute sample period over 30 days with ~5% margin

4. Start datalogger

5. After completion, open SD card contents using PC and verify 3000 individual measurements exist for each channel with associated time recorded

Success Criteria: 3000 data points exist for each channel with associated timestamp

Failure Criteria: Any other outcome results in test failure

Battery Life Test Procedure

1. Ensure battery is fully charged

2. Configure UUT for 15-minute sample period

3. Configure UUT for 3000 samples

4. Attach two temperature and two moisture sensors Note: test does not depend on recorded sensor data
5. Place datalogger in a secure location indoors, record start time, and start datalogger

6. One month later, recover datalogger and analyze data on SD card

Success Criteria: Minimum of 2,880 individual data points exist for each channel with timestamp

Failure Criteria: Fewer than 2,880 individual data points on any channel or timestamps fail at any point in the test

Lab Test Procedure

1. Configure DAQ for 15-minute sample period

2. Attach one sensor each of moisture and temperature

3. Attach one reference resistor and one reference capacitor to the remaining channels

4. Charge DAQ battery to full

5. Place DAQ in secure indoor location and leave for one month

6. After 1 month, retrieve and open the contents of the SD on a PC

Success criteria: Data exists for entire 1-month period, reference values measure consistently throughout test

Failure criteria: Battery does not last 1 month, measurement of reference values differ by more than 1 LSB over test, or sensor measurements change drastically

Environmental Test Procedure

1. Configure DAQ for 15-minute sample period

2. Attach two sensors each of moisture and temperature

3. Charge DAQ battery to full

4. Place DAQ in secure outdoor location with sensors buried

5. Start DAQ

6. After 1 month, retrieve and open the contents of the SD on a PC

Success criteria: Data exists for entire 1-month period if temperature was moderate, two weeks if temperature was consistently under 30 degrees

Failure criteria: Battery does not last for required duration or sensor channels measure significantly different

6. Budget

Our budget goal was $500.00 for both semesters of senior design for any PCB revisions and the final designed product. The final system is cheaper than existing commercial solutions on the market. The cost to build one sensor is $18.95, and the cost to build one DAQ is $77.07, including parts and shipping.

7. Previous Designs

7.1. Previous DAQ Designs

Our final data acquisition system is the product of many hours of testing, designing and fabricating various prototypes. Our design process began with the whiteboard. We decided what kind of micro-controller would work best, what kind of battery to use, how to communicate with our sensors, etc. After agreeing on a basic design we moved to the breadboard to test the design we created.

Once the breadboard was complete we moved our design to a perfboard to reduce the amount of noise we saw in our measurements and make our system more permanent. This lead to our first real prototype shown below. The prototype was used to confirm we could reliably record and store measurements onto an SD card with the use of our microcontroller. However, the first prototype would not allow us to measure capacitance (for moisture content) from the sensors we were using at the time (additional circuitry for this was being worked on).

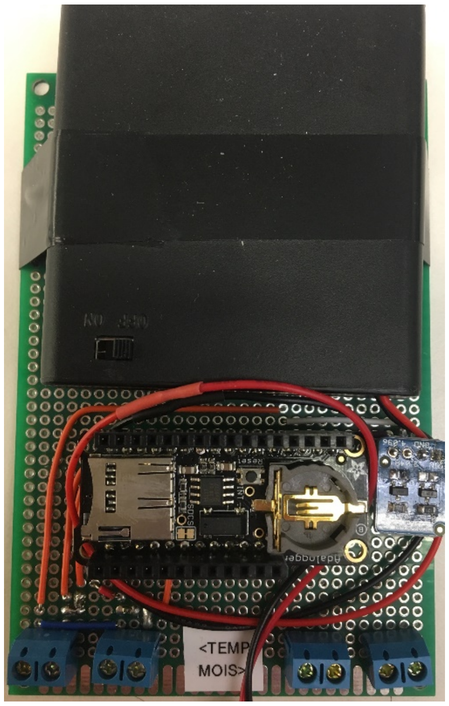


Figure 7: First DAQ Prototype

This initial prototype is what we would build on throughout the project.
Initial prototype strengths:

- Simple, scalable design to use in future prototypes
- Low power consumption
- Highly accurate temperature measurement

Weaknesses and areas to build upon:

- Power reduction (e.g. sleep mode) not implemented
- Capacitance measurement circuit to record soil moisture
- Weatherproofing
- Parasitic capacitance from cables interfering with measurement

Our second prototype was meant to be functional with our data logging prototype and measurement circuitry on a single PCB. This data acquisition system would be contained within a waterproof enclosure, with ports to the outside to reach the sensors. The PCB that was fabricated is shown below. The brains of this circuit will be the ATmega32u4 with an attached Adafruit Feather shield which hosts a micro SD card for data storage.

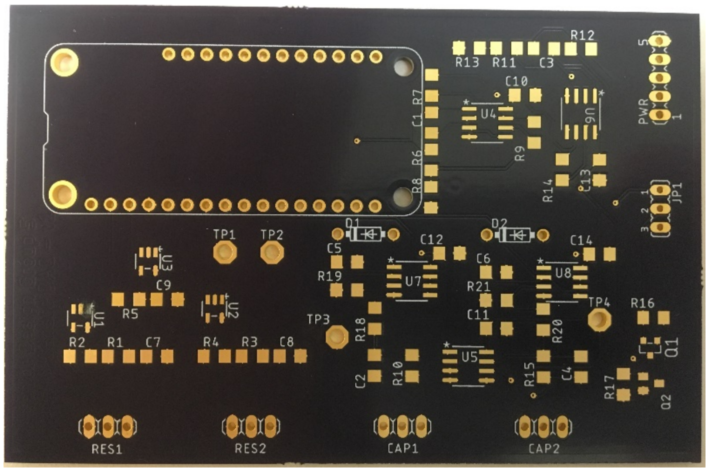


Figure 8: Second DAQ Prototype

This second prototype features:

- ATmega32u4 micro-controller
- Four independent sensor channels (two moisture, two temperature)
- Micro-SD card to log measurement data
- Real time clock to wake the micro-controller every 15 minutes to measure the state of the sensors
- The utilization of sleep mode to conserve battery life
- Battery pack containing four AA batteries

One of the biggest challenges of this project was creating the circuit to measure the small capacitance of the MEMS sensors that were provided. These sensors only varied a few picofarads over a wide range of soil moisture. The approach we used to measure this small capacitance was to design a circuit that applies a triangle wave to the capacitance sensor and then measures the current through that capacitor. The current is then converted to a voltage using a current to voltage converter with a square wave output. Using a peak detector we were able to continuously track the maximum amplitude and relate it to capacitance using the equation $C = \frac{I}{dV/dt}$. This was proven to work, and it was incorporated in the second prototype that is shown above.

Many of the challenges our group faced were the product of having sensors that were difficult to interface with. We were able to overcome all of these measurement challenges, but we later deemed the sensors to be unreliable for long-term use. As a result of this, our group decided to scrap the MEMS sensors we were originally working with and fabricate our own sensors. Creating our own moisture/temperature sensors would allow us to create sensors that are more accurate, reliable and easier to interface with.

7.2. Previous Sensor Design

Our group began researching various types of commercial moisture/temperature sensors in order to create our own. The first feasible design we found was a sensor fabricated on a PCB that used two fingers to measure the resistance of soil. The idea was that resistance would decrease as more water was added to the soil. In addition to the moisture sensor we would also have a temperature measurement circuit on the sensor. These sensors would have an on-board 3.3V reference, a buffer and two output channels (one for moisture, one for soil). The first version of our sensors is shown below.

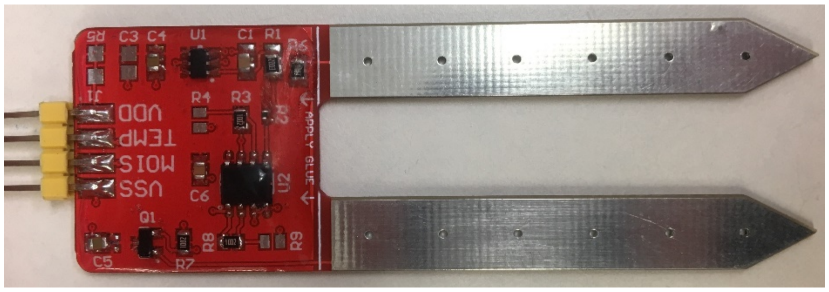


Figure 9: First Sensor

The sensor has two large plates which are measuring the resistance between the two to determine the soil moisture content. The circuitry is protected by a clear epoxy so the sensor can be submerged with no consequences.

The temperature measurement on the circuit works extremely well. Unfortunately, through testing we found that a resistance measurement for soil moisture was not reliable. The resistance would drift over time. Upon research we found that this is usually an issue with resistance soil measurements because current may not always choose the same path to flow through the soil, resulting in false measurements.

Appendix A: User Manual

1.0 – Purpose and General Information

The purpose of this user manual is to describe the setup and use of our data acquisition system (DAQ) for soil monitoring that we designed during the Fall 2018 Semester of EE 492. The DAQ is contained on a single two-sided printed circuit board (PCB). The PCB also has a custom enclosure design to protect the circuity on the PCB in an outdoor environment. The enclosure is constructed of delrin plastic and contains a gasket for water-proofing.

1.1 – System Overview

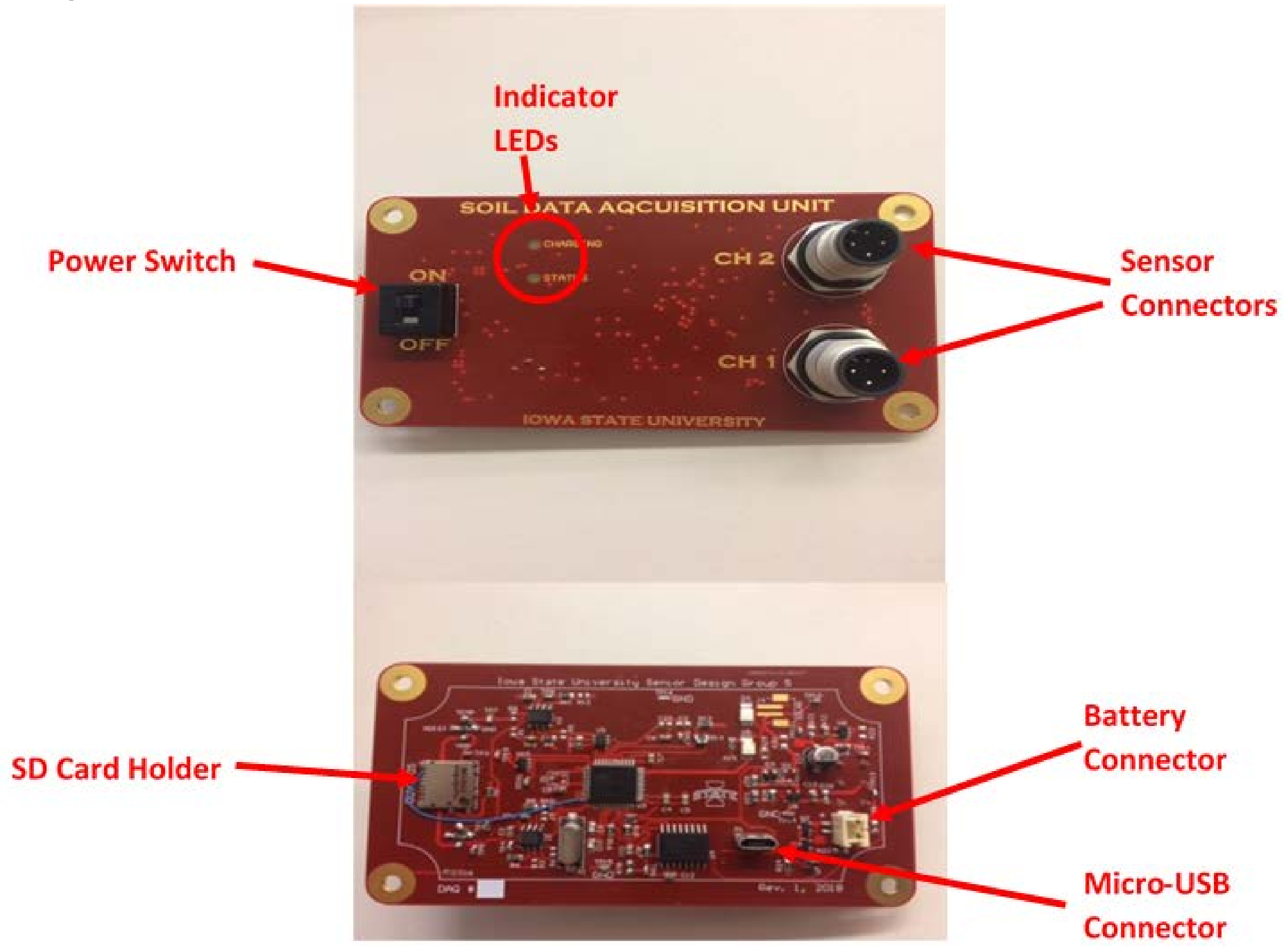


Figure 1: Top and bottom views of PCB with key features labeled

2.0 – System Features

Table 1 below summarizes the function and purpose of key features labeled in Figure 1.

Feature	Purpose
Power Switch	Connect and disconnect battery power to DAQ system. Must be switched to 'ON' position for DAQ to function.
Indicator LEDs	Two LEDs labeled 'CHARGING' and 'STATUS' indicate various states the system is in. 'CHARGING' LED will glow red when battery is being charged. The 'STATUS' LED will turn on and off to indicate different system conditions. The 'STATUS' LED should only turn on and off once when system powers up.
Sensor Connectors	The sensor connectors provide a connection for the DAQ to received data from the cables connected to external sensors placed in soil.
SD Card Holder	Provides a connection for the micro SD card. Temperature and moisture data is stored on the micro SD card along with other data.
Battery Connector	Provides a connection for battery voltage to power the DAQ PCB.
Micro-USB Connector	Provides a connection for recharging the system battery.

Table 1: Summary of Key Features

2.1 – Setup

This section describes the necessary steps to setup the DAQ system for use. It is assumed that the necessary software has been uploaded to the microcontroller on the DAQ already.

1) Put SD card into the SD card holder for data storage.

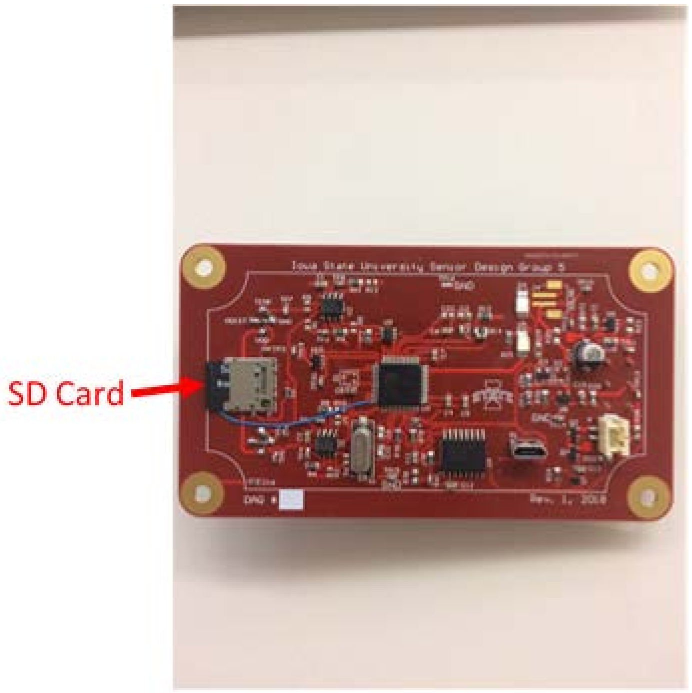


Figure 2: SD card inserted into DAQ

2) Connect battery to battery connector on PCB. The polarity of the battery is indicated below.

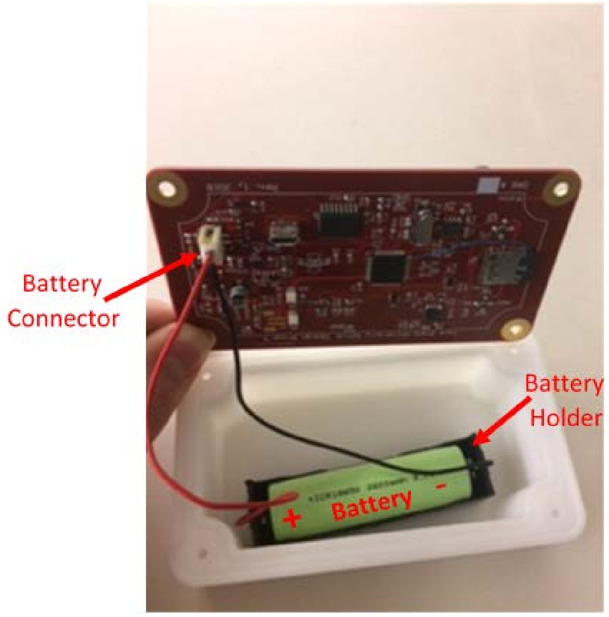


Figure 3: Battery polarity and connection to DAQ

3) Fully charge the battery by connecting the micro-USB connector to a computer. The ‘CHARGING’ indicator LED will glow red if charging and will turn off once fully charged.

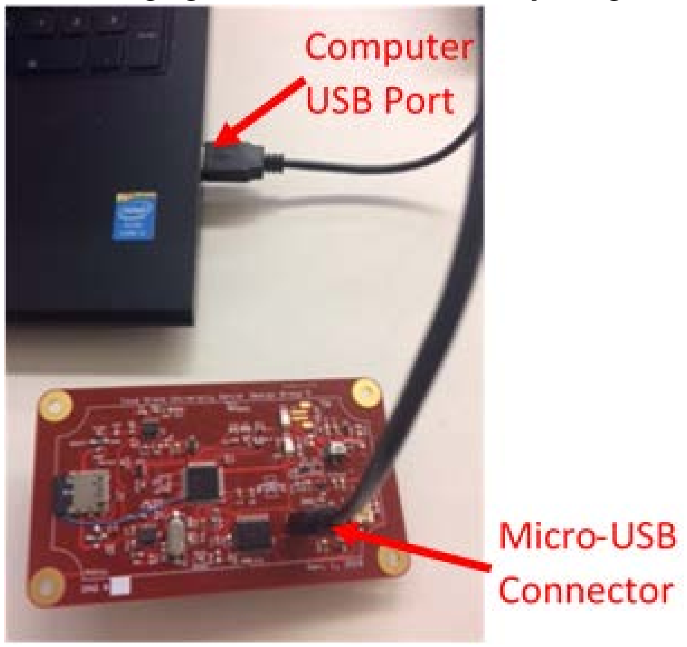


Figure 4: Recharging DAQ via micro-USB cable

4) Line up DAQ PCB on top of enclosure. There should be a snug fit as the top side of the PCB serves as protective lid while the bottom side with circuitry will be inside the enclosure.

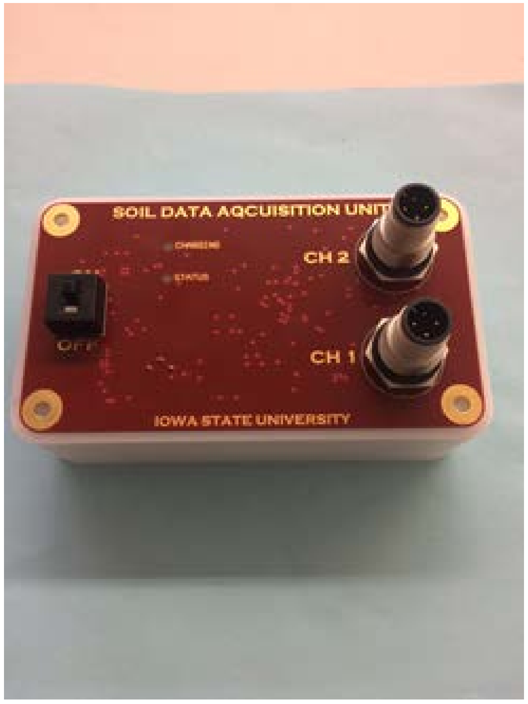


Figure 5: DAQ PCB placed on top of enclosure

5) Apply 4 corner thumb screws. Rotate clockwise to tighten.

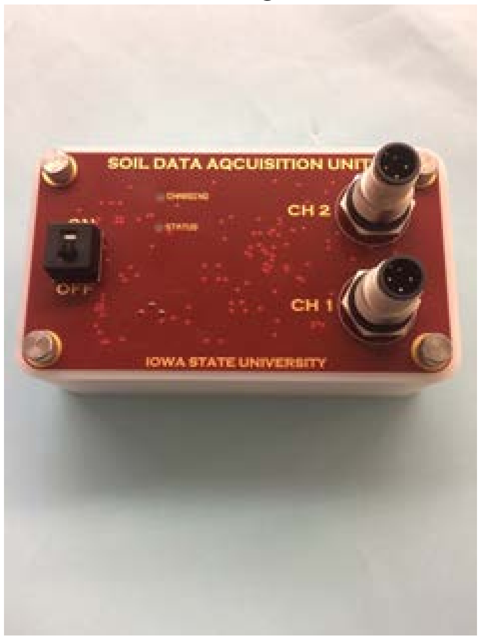


Figure 6: DAQ assembly with thumb screws installed

6) Attach cable connectors to the "CH 1" and "CH 2" sensor connectors on the PCB. Please note the alignment of the groove on the cable connectors and the notch on the sensor connectors of the PCB and rotate clockwise for a snug fit. Finally, turn on the power to the DAQ by pushing slider on power switch all the way to the top in the 'ON' position.

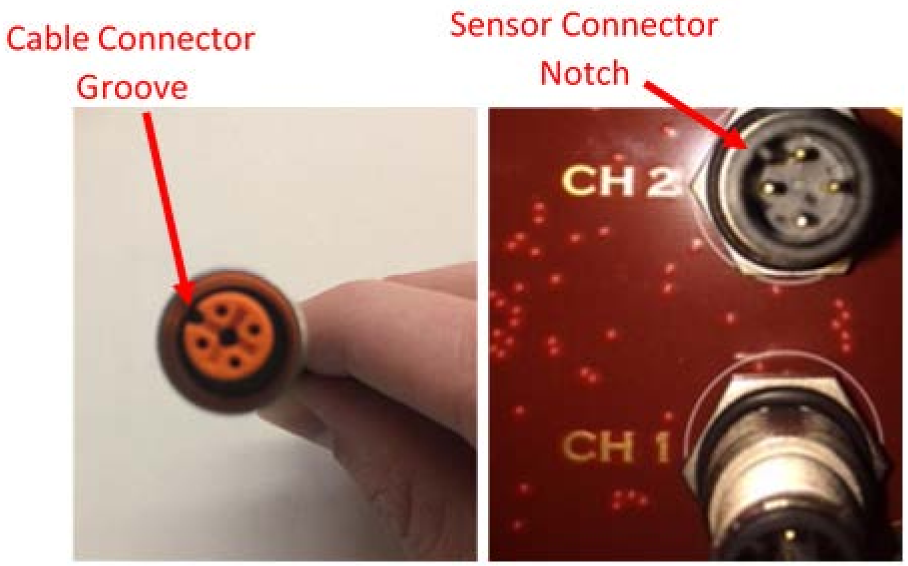


Figure 7: Cable groove and sensor connector notch to be aligned

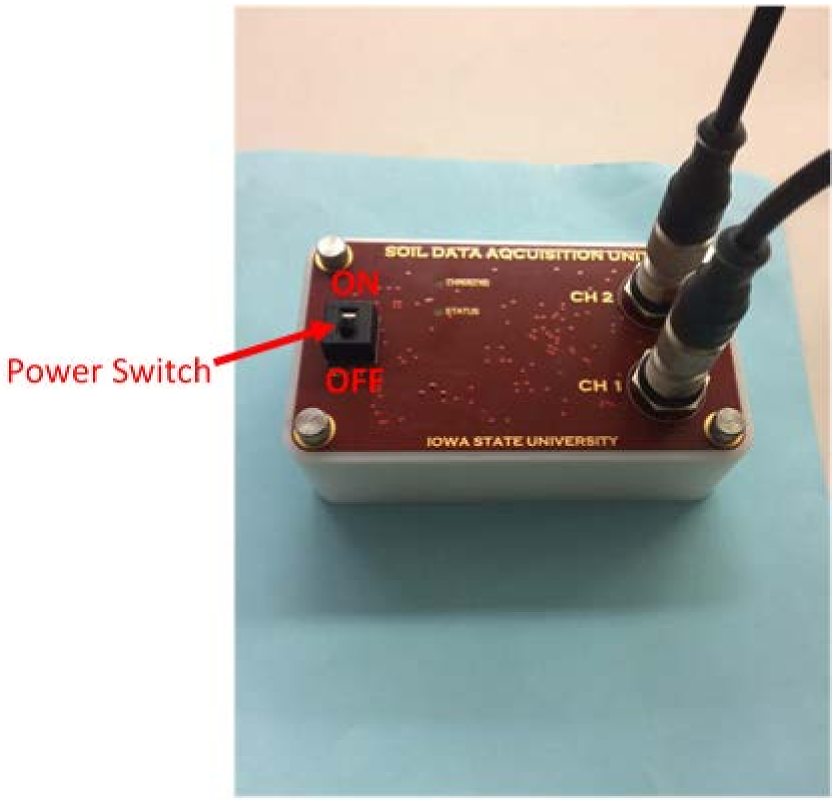


Figure 8: Power ON/OFF switch positions for DAQ

2.2 – Extracting Data

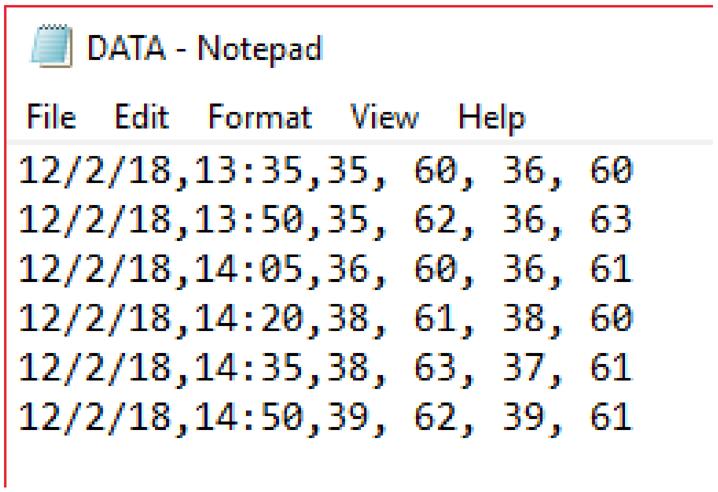
Once the DAQ has been properly setup, powered, and given time to run its data collection program, the data stored on the SD card can be extracted and analyzed on a computer. This can be done by using any computer that has Microsoft Excel for data analysis and an SD card reader to import the collected data into Excel.

1) Power off the DAQ by sliding the power switch to the 'OFF' position.
2) Remove the 4 thumb screws by rotating counterclockwise.

3) Remove micro SD card from the card holder and plug into a computer. An SD card adapter may be needed if the computer does not have an SD card reader of appropriate size (see below image).

Figure 9: Optional SD adapter for data extraction

4) Navigate to the SD card device on your computer to verify data has been collected. File Explorer can be used to do this if using a Windows operating system. A .txt file titled DATA.txt should be stored on the SD card. The data will be in comma separated value columns where the first two columns denote the timestamp of when the data sample was collected. The last 4 columns correspond to the temperature reading in degrees Fahrenheit for channel 1, the moisture content percentage for channel 1, temperature reading for channel 2, and the moisture content for channel 2 respectively.



```
12/2/18,13:35,35, 60, 36, 60
12/2/18,13:50,35, 62, 36, 63
12/2/18,14:05,36, 60, 36, 61
12/2/18,14:20,38, 61, 38, 60
12/2/18,14:35,38, 63, 37, 61
12/2/18,14:50,39, 62, 39, 61
```

Figure 10: Example time-stamped CSV data format

5) Open a blank Excel workbook and click on ‘Data’ ribbon. Then click on the Get Data drop down. Next drag mouse to ‘From File’ and then drag mouse to ‘From Text/CSV’ and click on it. Navigate to the DATA.txt file location and click on import at the bottom of the window.

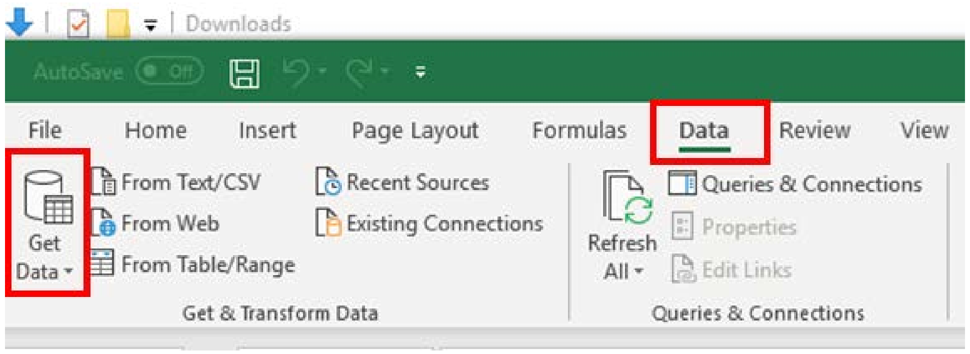
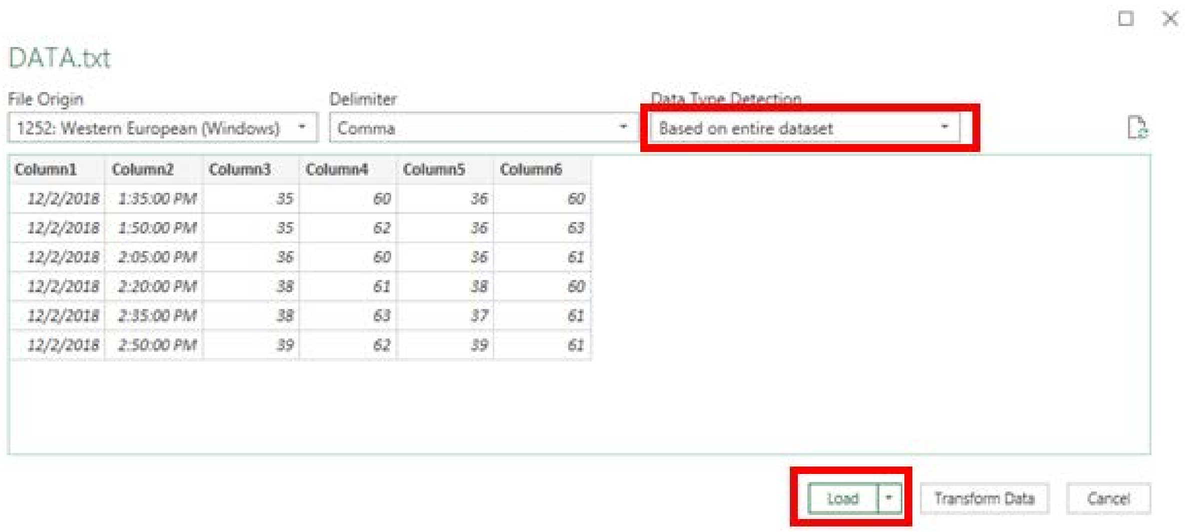


Figure 11: Excel 'Data' ribbon and 'Get Data' button

6) Specify how you would like the data to be formatted in the workbook in the DATA.txt window. The initial settings are okay except change the 'Data Type Detection' field to 'Based on entire dataset'. Then click 'Load' to load the data into the workbook.



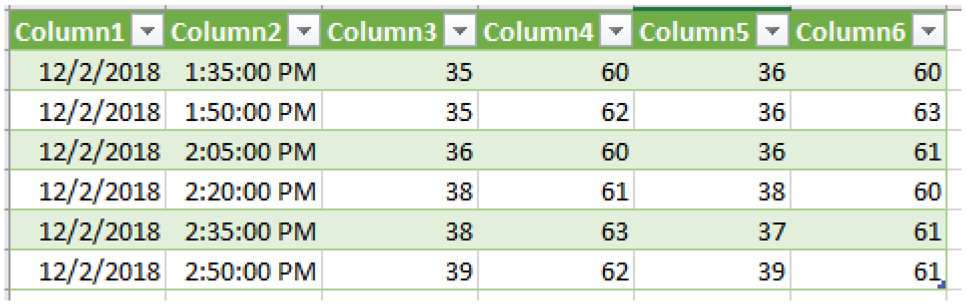
DATA.txt

File Origin: 1252: Western European (Windows)
Delimiter: Comma
Data Type Detection: Based on entire dataset

Column1	Column2	Column3	Column4	Column5	Column6
12/2/2018	1:35:00 PM	35	60	36	60
12/2/2018	1:50:00 PM	35	62	36	63
12/2/2018	2:05:00 PM	36	60	36	61
12/2/2018	2:20:00 PM	38	61	38	60
12/2/2018	2:35:00 PM	38	63	37	61
12/2/2018	2:50:00 PM	39	62	39	61

Figure 12: Excel DATA.txt window

7) Once the data is imported into the workbook, the final result should look similar to the figure below. Data analysis and various statistics can then be calculated as user desires.



Column1	Column2	Column3	Column4	Column5	Column6
12/2/2018	1:35:00 PM	35	60	36	60
12/2/2018	1:50:00 PM	35	62	36	63
12/2/2018	2:05:00 PM	36	60	36	61
12/2/2018	2:20:00 PM	38	61	38	60
12/2/2018	2:35:00 PM	38	63	37	61
12/2/2018	2:50:00 PM	39	62	39	61

Figure 13: Final data imported into Excel workbook for data analysis

2.3 - Maintenance and Debugging

Common things to debug are making sure all connections are satisfied as described in section 2.1 of this document, specifically, the SD card being placed into the holder, battery connection is properly made, and sensor connectors are properly connected with sensor cables. The primary maintenance concern is making sure the battery remains adequately charged. It is good practice to make sure the battery is fully charged before being placed out into field conditions, i.e., recharge it while data analysis is performed for a dataset and then once fully charged place back into the field. In cold conditions the battery will not perform as well and lose charge more quickly than when it is in more favorable conditions.

Appendix B: Assembly Manual

1.0 – Purpose and General Information

This guide serves as a reference to reproduce the DAQ systems delivered Fall 2018. It is intended for individuals with a moderate understanding of electronics and electronics assembly skills. Assembling this system with no understanding of electronics is not recommended. Covered topics include ordering PCBs, assembly, and firmware installation.

2.0 – Ordering PCBs

The bare PCBs can be acquired from any of hundreds of PCB manufacturers around the world. JLCPcb.com was the supplier of the DAQs delivered to the customer and provided good quality at a low-cost and short shipping time. Upload the PCB gerbers (delivered to customer in a .zip file) to the manufacturer of your choice. Below are the specific features you should quote from your PCB vendor.

- Dimensions: 2.504” x 4.512”
- 4 layer stackup
- 1.6mm thick
- Enig (gold) or Nickel plating acceptable
- Any soldermask color acceptable

Should any design changes be required, the original ECAD files for the DAQ can be acquired from the public project below and edited within Altium CircuitMaker (free account required).
https://circuitmaker.com/Projects/Details/Nathan-Coonrod-2/Soil-DAQ/

3.0 – Assembly Instructions

Ensure you have acquired all components listed in the bill of materials prior to beginning assembly. Additionally, you will require the following items/tools.

1. Fine tweezers
2. Heat pen or reflow oven
3. Solder paste stencil
4. Small squeegee for solderpaste
   Note: Old credit card works well
5. Solder paste
   Note: Available from ETG
6. Soldering Iron
7. Solder
8. Arduino bootloader to install bootloader on DAQ

Begin by placing the PCB to be assembled face down on a work surface so that the component pads are visible. Tape the PCB to the work surface to prevent it from shifting. Align the holes in the stainless steel stencil with the pads of the PCB. Tape the stencil in this position to prevent it from shifting. If the stencil is properly aligned, you should only be able to see the PCB pads through the stencil and no soldermask.

Once the stencil is properly aligned, apply a small bead of solder paste to the stencil and use the squeegee to distribute it evenly over all pads. Continue adding solder paste and distributing it with the squeegee until no pads are visible and there is a uniform layer of solder paste. Carefully remove the solder stencil without smearing the applied paste.

Begin placing components onto the PCB using small tweezers. Use your judgement when choosing the order to place components. It is usually wise to place the small components which are very prone to shifting last.

Once all components are placed, use a heat pen set to a low to moderate airspeed and a temperature of 350 degrees Celsius to begin reflowing the individual components. It will typically take only a few moments of heating with the pen for the solder to change from a dull grey color to a shiny silver at which point the solder has been reflowed and heat can be removed. Alternatively, a reflow oven can be used in lieu of a heat pen.

After all components have been successfully reflowed, inspect the PCB for any parts which have shifted off their pads, solder bridges, or other defects. Use your judgment as well as a heat pen, soldering iron, and tweezers to rework any issues. Once the PCB is free of all defects, place and solder all large or through-hole components not placed earlier.

With all components soldered and free of defects, power the circuit for the first time using a power supply with current limiting set below 1A to protect the circuit in case of any unknown faults from assembly. After verifying there are no shorts or solder bridges, connect a battery to the DAQ as well as a micro USB cable. Whenever a battery is connected for the first time, use a DMM to measure the voltage on the gate (pin 1) of both Q1 and Q2. If the battery protection circuitry deems the battery safe to operate, these nodes will both be the battery voltage with respect to DAQ ground (not battery ground). The charging indicator LED should light up after the micro USB lead is connected and begin charging the battery. If the battery is fully charged, the LED may not light.

4.0 – Burning the Bootloader

At the core, the DAQ system is an Arduino using an ATMEGA32u4 microcontroller running a bootloader for an Adafruit Feather 32u4. As such, the bootloader must be loaded prior to installing the DAQ software itself. Think of this as installing the operating system prior to installing the end program needed by the user.

Once the hardware has been verified, the bootloader and software must be flashed onto the DAQ. Install the board support package for the ATMEGA32u4 by installing Adafruit's board library. The DAQ can be treated as equivalent to an Adafruit Feather 32u4. To install the Feather 32u4 bootloader, use the following guide and specifically the section titled "Connecting Arduino as ISP to Target."

https://learn.sparkfun.com/tutorials/installing-an-arduino-bootloader/all

Pin Mapping to Install Bootloader	
Pin on Arduino UNO	Location on DAQ
VCC	TP12
GND	Any GND test point
MOSI (D11)	J3 pin 3
MISO (D12)	U3 pin 11
SCK (D13)	J3 pin 5
D10	U3 pin 13

Table 1: Map Between Pin Name and Location on PCB

The guide above provides the specific steps required to connect the Arduino UNO to the listed points above and flash the bootloader. Once the bootloader is flashed, Arduino software can be written to the DAQ as if it were an Arduino. Open the DAQ software in the Arduino IDE, select the correct serial port, and hit upload. Upon successful upload, the DAQ is ready for operation once an SD card is inserted. Note that in order to correctly set the internal time of the RTC, the current time must be entered to the variables at the top of the DAQ program before uploading to the DAQ. Once the correct time is uploaded, the battery cannot be unplugged without losing the current time.

5.0 – Installing DAQ Firmware

Once the bootloader has been successfully installed in section 4.0, the user can upload code as if the DAQ were a normal Arduino over USB connection. Due to the specific requirements of this project and how the 32u4 microcontroller works, there are some specific steps which may be required to successfully install the Arduino sketch.

A 'fresh' DAQ which has just the bootloader can be programmed by simply connecting to a host PC, choosing the correct serial port in the Arduino IDE, selecting USBISP as the programming interface, and uploading the desired sketch. However, if the board has been previously programmed with DAQ software, additional steps are required. Because the DAQ is designed to be in a deep sleep mode for the majority of its life, it is difficult to establish serial communication with it to overwrite the existing program. When the DAQ does wake to take measurements, it does so for only ~50ms which is far too short to establish a serial connection and be programmed. So to program a DAQ which is asleep, the user must force it out of sleep. A procedure for this is detailed below:

1. Power on the DAQ with a battery connected
2. Plug in a USB cable to a host PC with the Arduino IDE and the DAQ software
3. Using a bare wire, short pin U3.13 to any of the GND testpoints on the board
   Note: An easy way to do this is to use a multimeter in current measurement mode and short the points above by probing ground and U3.13
4. Release the probe from U3.13 and monitor the available COM ports listed in the Arduino IDE, after several seconds you should see a new port listed, select it.
5. By the time you've selected the port, the DAQ has booted and gone back to sleep, but you now know the proper serial port. So short U3.13 once more and press upload immediately after releasing the probe

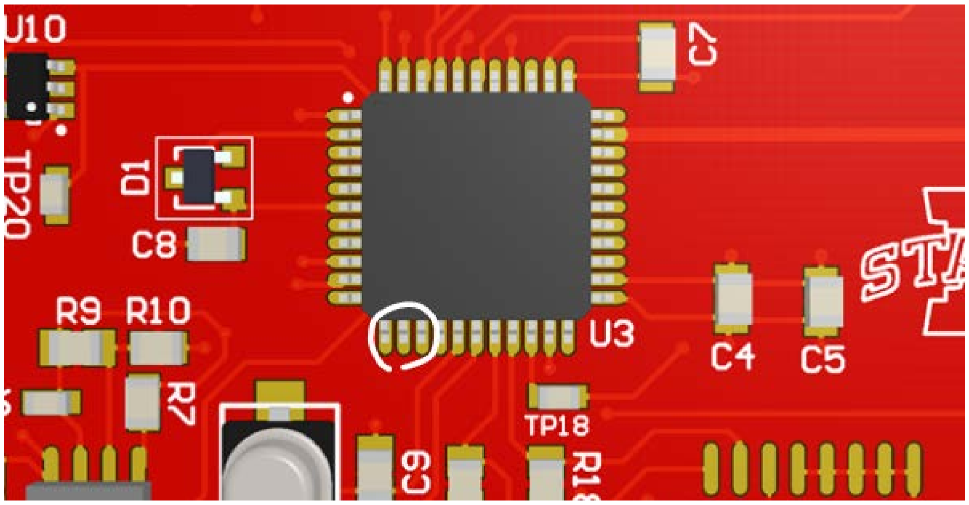
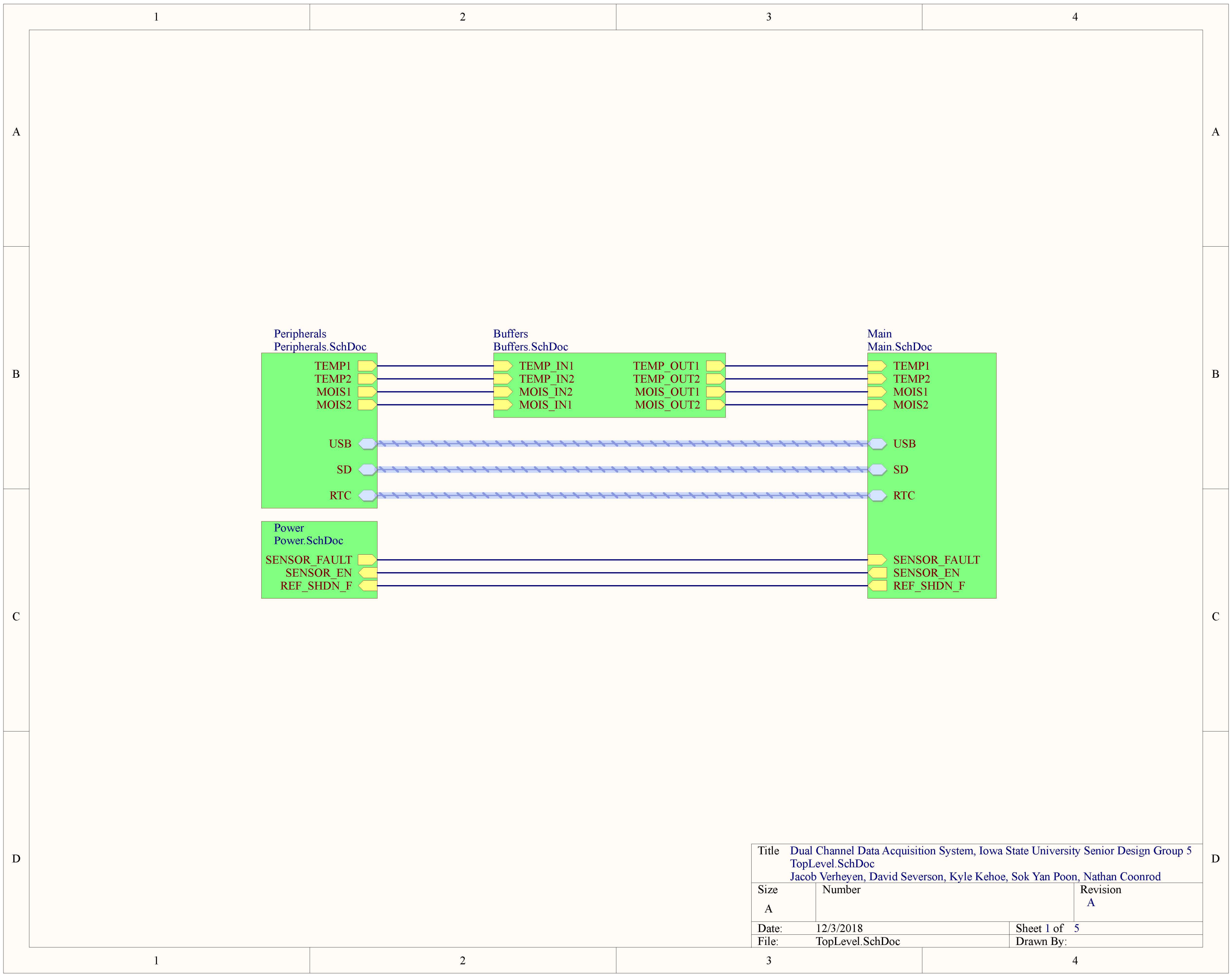
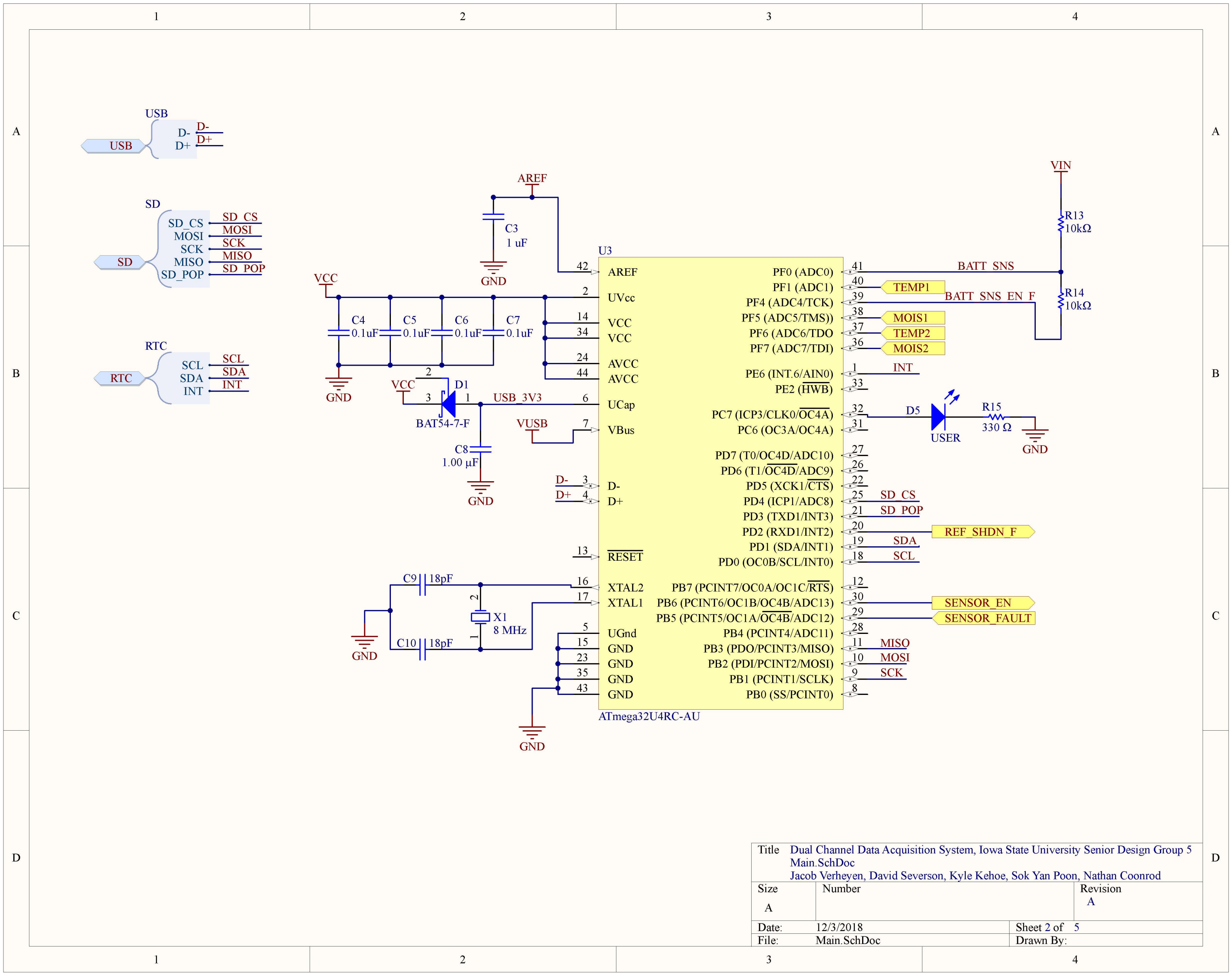


Figure 1: Probe location detail

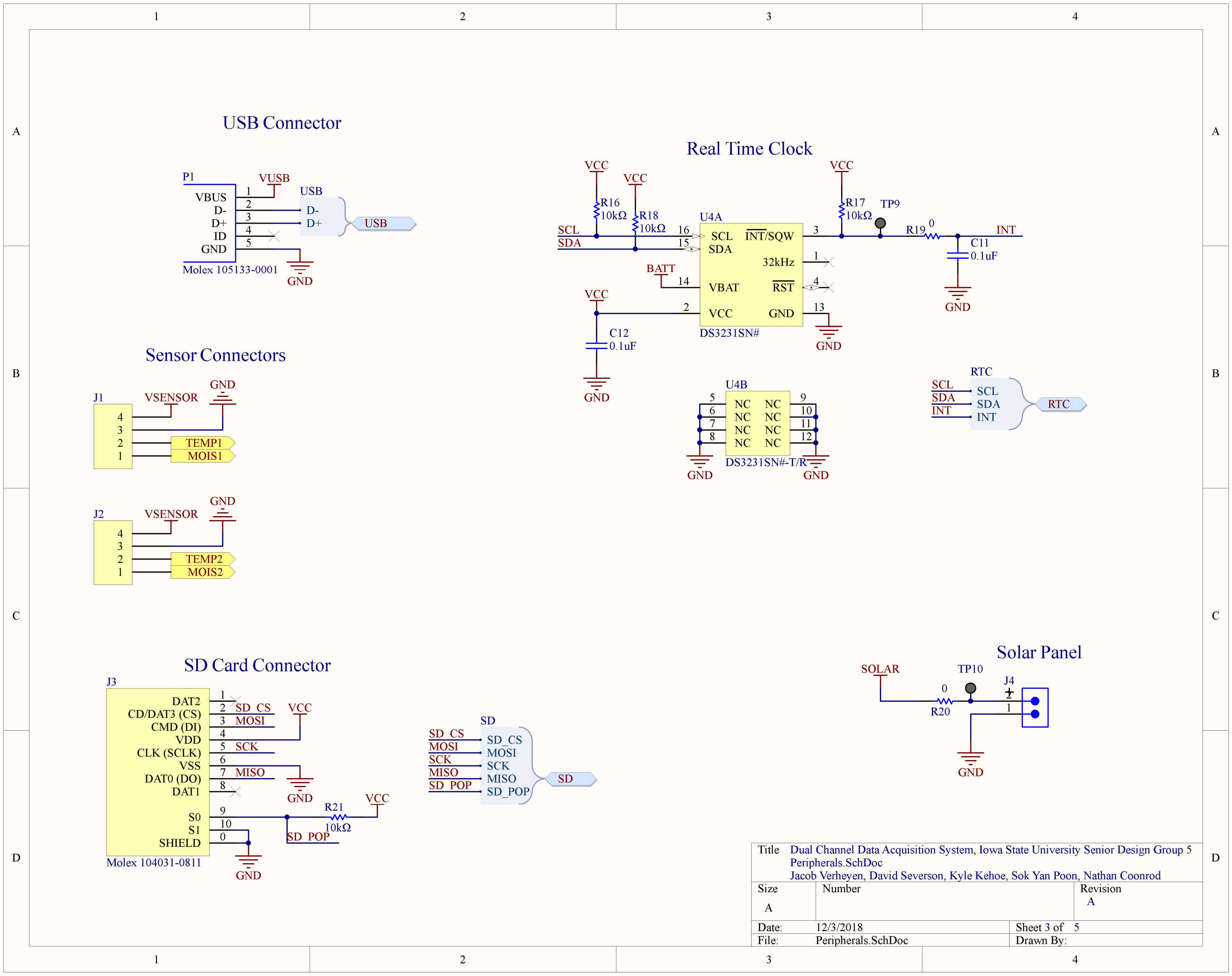
Appendix C: Schematics



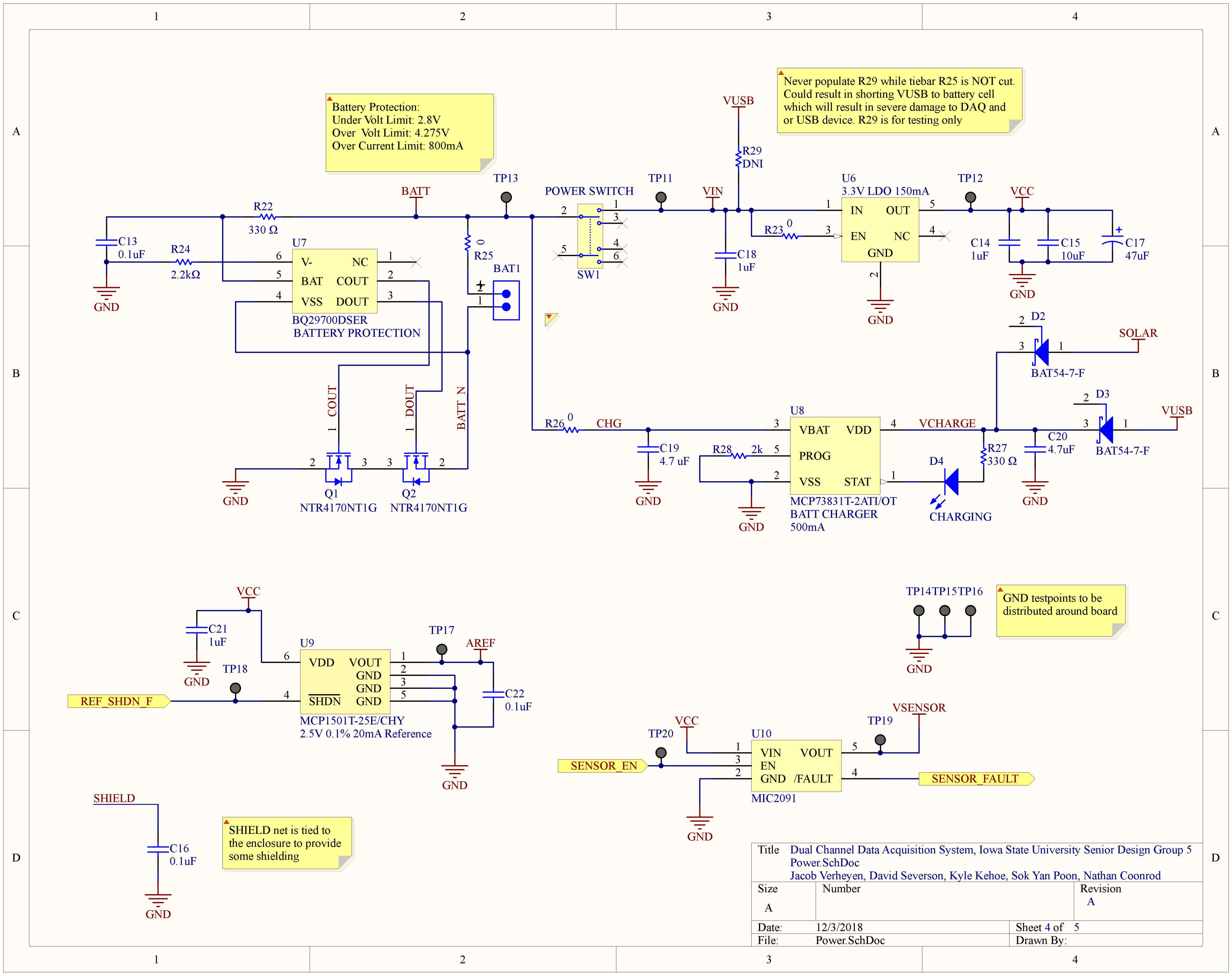
Title	Dual Channel Data Acquisition System, Iowa State University Senior Design Group 5 TopLevel.SchDoc Jacob Verheyen, David Severson, Kyle Kehoe, Sok Yan Poon, Nathan Coonrod		
Size A	Number		Revision A
Date:	12/3/2018	Sheet 1 of 5	
File:	TopLevel.SchDoc	Drawn By:	



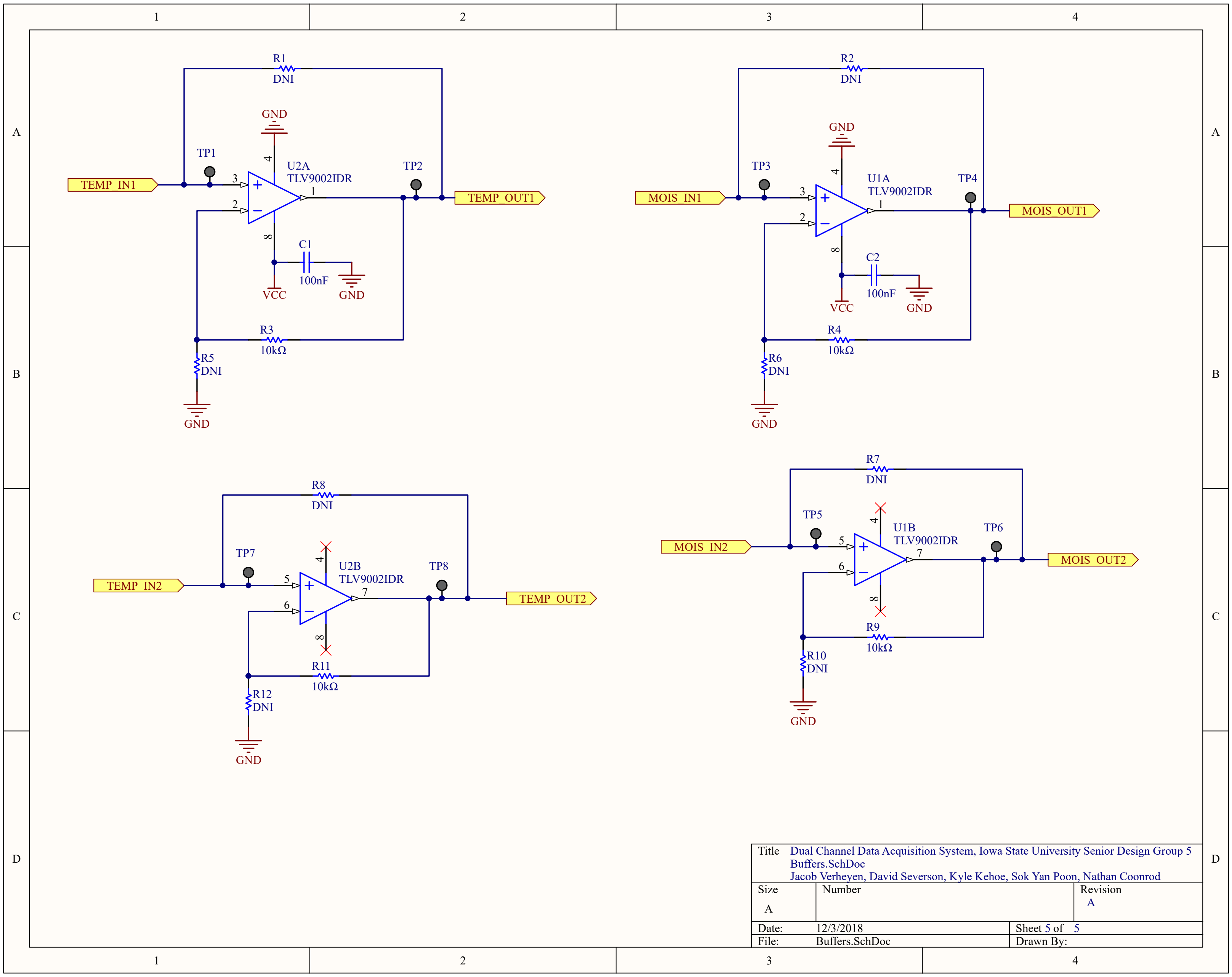
Title	Dual Channel Data Acquisition System, Iowa State University Senior Design Group 5 Main.SchDoc Jacob Verheyen, David Severson, Kyle Kehoe, Sok Yan Poon, Nathan Coonrod	
Size A	Number	Revision A
Date: 12/3/2018		Sheet 2 of 5
File: Main.SchDoc		Drawn By:



Title	Dual Channel Data Acquisition System, Iowa State University Senior Design Group 5 Peripherals.SchDoc Jacob Verheyen, David Severson, Kyle Kehoe, Sok Yan Poon, Nathan Coonrod	
Size A	Number	Revision A
Date: 12/3/2018		Sheet 3 of 5
File: Peripherals.SchDoc		Drawn By:



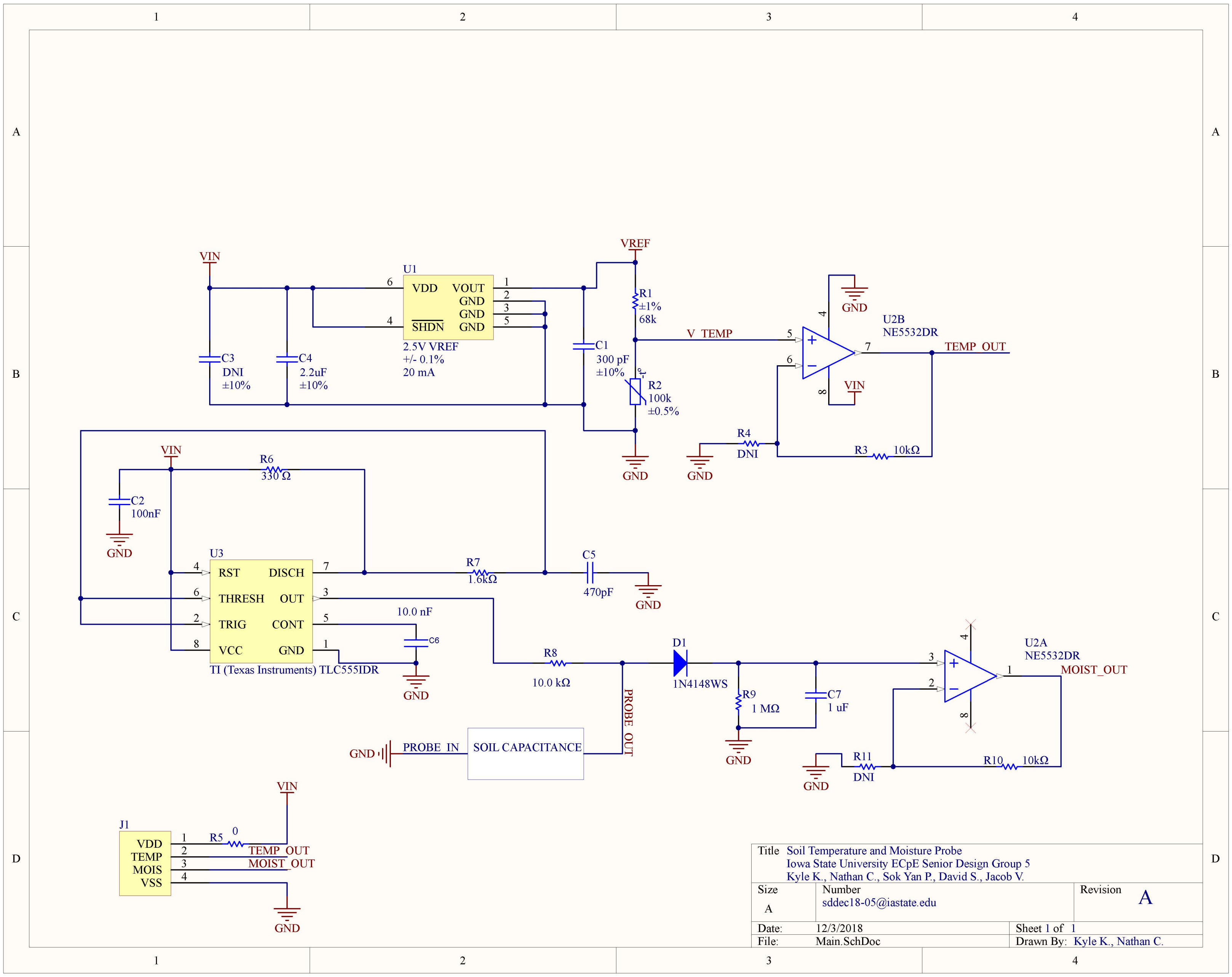
Title	Dual Channel Data Acquisition System, Iowa State University Senior Design Group 5 Power.SchDoc Jacob Verheyen, David Severson, Kyle Kehoe, Sok Yan Poon, Nathan Coonrod		
Size A	Number		Revision A
Date:	12/3/2018	Sheet 4 of 5	
File:	Power.SchDoc	Drawn By:	



Title	Dual Channel Data Acquisition System, Iowa State University Senior Design Group 5 Buffers.SchDoc Jacob Verheyen, David Severson, Kyle Kehoe, Sok Yan Poon, Nathan Coonrod

Size	Number	Revision
A		A

Date:	12/3/2018	Sheet 5 of 5	
File:	Buffers.SchDoc	Drawn By:	



Title Soil Temperature and Moisture Probe
Iowa State University ECpE Senior Design Group 5
Kyle K., Nathan C., Sok Yan P., David S., Jacob V.

Size	Number	Revision
A	sddec18-05@iastate.edu	A

Date: 12/3/2018 Sheet 1 of 1
File: Main.SchDoc Drawn By: Kyle K., Nathan C.