LOW ENERGY WIRELESS SOLUTION FOR SOIL MOISTURE MONITORING

By

JAKUB KOLODZIEJSKI

A thesis submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Dr. Yanyong Zhang

And approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October 2018

ABSTRACT OF THE THESIS

Low Energy Wireless Solution for Soil Moisture Monitoring

by Jakub Kolodziejski

Thesis Director: Dr. Yanyong Zhang

With a growing world population placing ever increasing demands on agriculture and an increasing scarcity of water suitable for irrigation, there is a need for efficient irrigation techniques utilizing comprehensive knowledge of soil water content. While commercial solutions are available that provide the required of information, none were found that are low cost, low maintenance, and high durability. This thesis presents a sensor and data backhaul design that addresses these three key parameters with measurable results. The design uses a TPIP-K transmit-only wireless platform for remote reporting of the measurement data. Since soil moisture measurements need only to be taken at regular but not precise intervals and individual data points are not critical by themselves, no bi-directional communications are necessary, leaving communication channels open to measurement data only and allowing more sensors to be deployed for a given portion of wireless spectrum. Additionally, since the most energy consuming component is the wireless radio itself, the transmit-only design lends itself to long battery life because the radio is only active during the brief times data needs to be reported. A long battery life coupled with wireless communication means deployment is simple and unobtrusive to other agricultural procedures. The design of the moisture sensor itself is based on a variable

value capacitor whose dielectric material is the water in the soil, making it simple and inexpensive to manufacture. The dielectric constants of air ($\varepsilon_r = 1$) and water ($\varepsilon_r = 80$) have a large enough difference that the concentration of water per unit volume in the soil will correspond to a specific capacitance value of the sensor. Making use of hardware features specific to the MSP430 microprocessor on the TPIP-K, precise and repeatable measurements can be made while maintaining low energy usage. Individual sensor calibration for a particular soil is achieved with measurements taken at low and new battery voltages, to account for battery aging, and approximately 25-30% and 100% field capacity, the point at which any additional water is drained from the soil by gravity and made unavailable to plants. Using bilinear interpolation based on the calibration data, the sensor can than report the volumetric water content of the soil between the calibration points to better than 3%, which is the range and detail of information most useful for agricultural irrigation purposes. The resulting sensor solution meets the set design goals of developing an inexpensive, low maintenance, and robust soil moisture sensor.

Table of Contents

1 Introduction

Most of the Earth's surface is covered in liquid water. In total, Earth has roughly 1.39 x 10^{21} liters of water spread out most over oceans, lakes, glaciers, rivers, and clouds [1]. However, less than 1% of that exists as fresh liquid water [2]. Pollution of fresh water sources has led to issues with water for drinking as well as irrigation [3]. An increasing number and severity of droughts has also shrunk the usable water supply in many areas of the world [4]. Lack of access to clean water for irrigation naturally has a significant negative impact on crop yields which is a serious problem for a world population of over 7.4 billion and counting [5]. While humanity figures out ways to undo the damage done to the environment, it must also be careful about using the ever-shrinking water resources still available.

To that end, the use of smarter and more efficient irrigation systems and techniques is necessary. Such a solution depends on having detailed knowledge about the ever-changing field soil moisture content. Too little water can reduce crop yield and even cause total loss of plants. Too much water can harm plants as well as being wasteful and contributing to increased fertilizer runoff, which is both costly and environmentally unsound. With an accurate picture of the soil moisture content, a smart irrigation system can be setup to provide water when and where it is necessary, even possibly maintaining the bare minimum soil moisture level required. To have such an accurate picture, a field may have hundreds or thousands of sensors installed to provide accurate highly localized information. A greater granularity in the soil moisture information allows for finer control of the irrigation system as well as the ability to detect faults which may yield to too much or too little water being supplied to the crops.

Many commercial solutions exist for monitoring soil moisture levels. However, there do not appear to be any available that provide a system which is inexpensive, durable, and easy to maintain while still providing a reliable level of accuracy. Such a system depends on components that can be cheaply yet reliably manufactured. In addition, the sensors must be durable enough to survive installation for extended periods of time in a harsh agricultural environment. Furthermore, little to no maintenance is an essential factor if hundreds or thousands of these sensors are to be installed in a field to provide the necessary high-resolution field data.

The structure of this thesis is presented as follows. First, it presents an overview of currently available technologies and techniques designed to address the problems of soil moisture content monitoring. Following this is an in-depth look into the proposed solution, providing implementation details as well as design tradeoffs and issues encountered during development. Lastly, experimental data is presented and analyzed demonstrating the accuracy and longevity of the low energy sensor and backhaul design.

2 Soil Moisture Sensing Technologies

There are several do-it-yourself (DIY) and commercially available sensing solutions specifically aimed at the task of measuring soil moisture. While some of these sensor technologies lend themselves to being inexpensive, they tend to suffer in terms of long term durability. On the other hand, the more durable and accurate measurement solutions tend to be comparatively expensive, require regular maintenance, or need trained personnel to operate. This section will discuss the most popular technologies in use along with each of their strengths and weaknesses.

2.1 Battery-style

This sensor technology is often used in inexpensive consumer sensors such as the Luster Leaf 1820 Rapitest Moisture Meter (Figure 2-2). The sensor probe is composed of two dissimilar metals separated by an insulator (Figure 2-1). When inserted into soil, the moisture present in the soil acts as an electrolyte between the two metals of the probe creating a crude battery. The voltage difference between the two metals varies with the moisture content of the soil and is typically measured by an analog voltmeter type display built into the top or handheld portion of the device. The displays have a seemingly arbitrarily defined scale from “dry” to “wet” or 0 to 10. These types of sensors are not meant to be installed for any length of time because they require wiping of the probe between readings due to electrolysis and oxidation of the probe metals.

Figure 2-1: Rapitest Moisture Meter

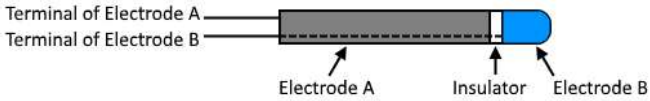

Figure 2-2: Battery-style moisture sensor

Experimental testing of the Rapitest sensor revealed measurement variations of 10-20% of the measurement scale in repeated readings depending on how tightly the soil was packed around the sensor, the type of soil, and whether the sensor was wiped before being reinserted in the soil. The initial peak value measured when the sensor was first inserted in the soil correlated with different levels of soil moisture content. However, the readings would subsequently decrease over the course of several seconds before stabilizing on a final value which did not vary with different levels of soil moisture. Potting soil that had been watered and subsequently allowed to dry over the course of a week did not change noticeably in this final value while visibly becoming drier. Placing the Rapitest sensor in tap water and then the same water filtered using an ordinary household carbon filter netted a 30% higher initial reading on average in the tap water versus the filtered water, indicating a strong effect of the water's salt and mineral content on the sensor readings despite the amount of water remaining the same. Although interfacing a microcontroller to this style of sensor would be trivial, requiring only a simple measurement using the microcontroller's ADC, the lack of measurement repeatability and inability to leave the sensor installed in the soil for an extended period makes it not a feasible choice for this application.

2.2 Resistive

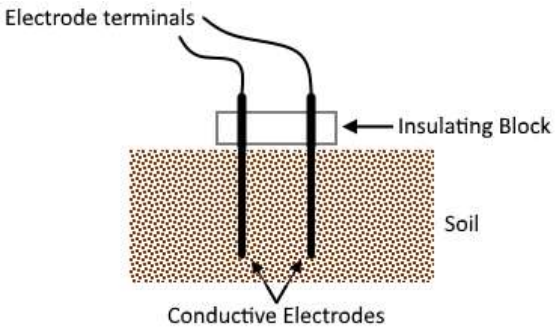

Figure 2-3: Resistive moisture sensor

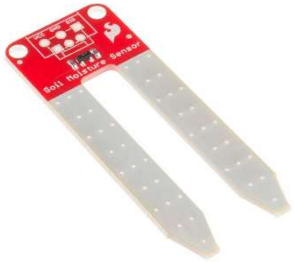

Figure 2-4: Sparkfun SEN-13322 soil moisture sensor

The resistance based sensor is quite popular with members of the DIY community interested in gardening and plant monitoring. These sensors can be readily purchased or easily made. The design is simply two metal electrodes spaced approximately 0.5 - 1cm apart that have an electrical current pass between them through the moist soil [Figure 2-3]. The higher the soil moisture content, the lower the resistance between the two electrodes, resulting in a smaller voltage drop. This voltage drop can easily be measured using a microcontroller's ADC. The two popular designs of this sensor are either two metal spikes made of stainless or galvanized steel or a two-pronged fork shaped circuit board with a large exposed copper area on each spike that was gold or tin plated to reduce corrosion [Figure 2-4]. In the case of the circuit board based sensors, salts from the soil dissolve in the water and corrode the electrodes relatively quickly because the copper base and gold or tin plating are very thin. The thicker metal rod electrodes of the latter design take a lot longer to corrode or corrode very little in the case of the stainless steel. However, there still is the problem of oxides forming and minerals bonding to the surface of the electrodes. To mitigate this, an AC current can be used for measurement instead of DC to prevent electrolysis of the electrodes. Additionally, the electrodes are sometimes embedded in a

gypsum block or gypsum based granular media as is the case with the WATERMARK sensor from Irrometer Company to reduce corrosion of the electrodes due to salts in the soil. The hydrophilic properties and uniformity of the gypsum media draw water from the soil and tend to keep an even distribution of the moisture around the sensor's electrodes aiding in measurement consistency [6].

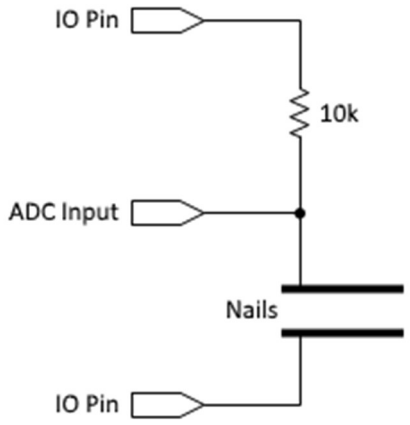

Figure 2-5: Electric Imp based sensor circuit diagram

In an experimental setup, a simple resistive type sensor was made using a pair of 12-gauge galvanized steel nails mounted 1cm apart as in Figure 2-3. The sensor was connected to an Electric Imp microcontroller using the circuit in Figure 2-5 and readings were taken by applying 3V across the sensor using the IO Pins for 100ms (to ensure reading stability) and the resulting voltage drop measured using the microcontroller's ADC input. Following a measurement, the current through the sensor was reversed for an additional 100ms in an attempt to counteract the electrolysis effect. There was no practical way of producing a high frequency AC signal (which would reduce the effects of electrolysis further) and performing the necessary measurement using this microcontroller platform.

Figure 2-6: Corroded resistive sensor probe

The sensor was installed in the soil of a potted house plant watered with ½ cup of filtered tap water consistently once a week in a room kept relatively stable at 22°C and 50%RH. The microcontroller performed a reading every 30 seconds. The sensor readings over the course of an hour were 5.6kΩ ± 16% before the plant's scheduled watering and 3.4kΩ ± 22% just after. With such inconsistency in readings, it was necessary to average many hours of data to even have an idea of the soil moisture content and whether it was increasing or decreasing. Additionally, the sensor's nominal resistance slowly increased over several months making determination of absolute soil moisture content impossible. After removing the sensor from the soil several months later, it was discovered that slight corrosion of the metal electrodes had taken place and hard mineral deposits had bonded to the surface which reduced the effective area of the sensor exposed to the soil and increase the measurable resistance of the sensor [Figure 2-6]. For this reason, a sensor with the electrodes embedded in gypsum would last longer. However, the gypsum itself will

naturally deteriorate due to exposure to water in the soil, eventually resulting in sensor failure as well [7].

2.3 Tensiometer

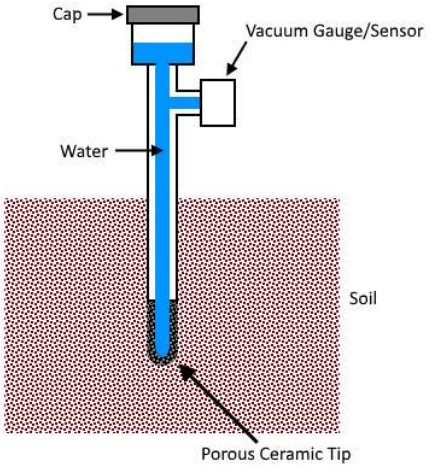

Figure 2-7: Tensiometer

These types of sensors are the "de facto" standard method of monitoring soil moisture. Unlike the other methods described, a tensiometer does not directly measure the absolute water content of the soil. Instead, it determines the relative availability of water to plants by how difficult it is for a plant to extract water from the soil via its root system eliminating the need to calibrate the instrument for a particular soil composition. A tensiometer is comprised of a tube filled with water with a sealed top and a bottom tip made of a porous material (Figure 2-7). The tube is inserted into a predrilled hole in the soil with the tip at the desired measurement depth. As the soil dries, osmotic pressure draws water out of the tensiometer through the porous tip and imparts a suction force in the tube which can be measured with a pressure gauge. The drier the soil, the greater this vacuum pressure is. When the soil is between the points of field capacity and saturation, no water is drawn

from the tensiometer and the pressure reading is zero. At water content below field capacity, water is drawn osmotically out of the tensiometer by the soil, causing a measurable decrease in pressure inside the tube.

Tensiometers, like those available from Irrometer Company, have a mechanical pressure gauge which can easily be replaced with an electronic pressure transducer suitable for interfacing with a microcontroller. Additionally, because only a pressure measurement is made and all parts of the sensor exposed to the soil are corrosion resistant (plastic and ceramic), the soil's salinity, pH, and mineral content have no direct effect on measurements or sensor longevity. The major downsides of using a tensiometer is that it requires regular maintenance (water refill and removal of air bubbles) and cannot be left permanently installed in areas where temperatures drop below 0°C which would cause the water inside to freeze and rupture the instrument.

2.4 Capacitive

These sensors in their simplest form consist of two insulated metal electrodes that use the water in the surrounding soil as a dielectric to form a variable capacitor. The dielectric constant of air is very close to 1, while that of water is 80 (at 20°C). The compounds in the soil itself have dielectric constants typically in the range of 3 to 15 [8], which leaves water as the most significant contributor to the value of this "soil capacitor". Higher water content in the soil will increase the capacitance value of the sensor. Because the electrodes of the sensor are insulated from the surrounding soil and its water, no corrosion can take place which allows for permanent installation in the ground and gives the sensor a very long lifetime in comparison to the battery and resistance type sensors.

The capacitance value is measured primarily in one of two ways. The first method uses the sensor's electrodes as part of an RC oscillator circuit generating an AC signal whose frequency depends on the sensor's capacitance and therefore the water content of the surrounding soil. Different frequencies will correspond to different levels of soil moisture. The second method measures the capacitance value of the sensor directly by measuring the charge or discharge time which corresponds to the capacitance value of the sensor. Due to the simplicity of the circuit and sensor design and the potential for long term installation in soil, this project focused on this style of capacitive sensor.

2.5 TDR

Time Domain Reflectometer (TDR) sensors work by measuring the effect of soil water content on the speed of an electromagnetic wave. Similar in design to capacitance based sensors, a pair of insulated electrodes is inserted into the soil to act as an electromagnetic wave guide. When an electromagnetic wave is sent down the wave guide, its speed is proportionally decreased with higher levels of water in the soil. The time it takes the wave to reach the base of the wave guide and reflect back is used to calculate the moisture content of the soil [9]. Commercially available solutions of this type are generally expensive due to the electronics involved.

2.6 Ground Penetrating Radar

Like TDR, ground penetrating radar (GPR) also measures the effect of soil water content on the speed of an electromagnetic wave. However, instead of the wave traveling down a wave guide and back again, it travels between a separate radar transmitter and

receiver. As the electromagnetic wave travels from the transmitter and passes through the soil, it is reflected back toward the receiver antenna. High soil water content will have a high dielectric constant and thus result in a slowing down of the electromagnetic wave. The difference between the expected arrival time of a wave for a known level of water content versus the actual arrival time is then used to infer the actual water content of the soil being measured. One of the biggest measurement characteristic differences between this method and those so far described is the ability to take measurements over a relatively large volume of soil, on the order of meters versus centimeters or less [10]. This reduces the effects of localized pockets of moisture and air on the overall reading. Unfortunately, radar is a very energy hungry and relatively costly technology, making it not suitable for a low-cost battery powered sensing system.

2.7 Neutron Probe

This is one of the more exotic methods of measuring soil moisture often used in research for its high accuracy. A radioactive pellet containing americium-241 and beryllium is lowered into a borehole in the ground [11]. The fast neutrons given off by the pellet slow down and scatter when they encounter the hydrogen atoms of the water molecules in the soil. The number of scattered slow neutrons is measured with an instrument to determine the absolute water content of the soil. Higher soil moisture levels would result in a greater number of scattered neutrons being detected. Neutron probes have several advantages; including being able to take measurements at different depths with a single probe setup (the radioactive pellet can be held at the desired height in the bore hole) and like GPR, the effective area of measurement is greater than that of the other sensor

types resulting in measurements that are less affected by localized moisture variations close to the sensor. These sensors, while very accurate once calibrated for a particular soil composition (to account for the neutron scattering caused by other elements in the soil), are expensive and, due to the use of bore holes, cannot be easily relocated [12].

3 Proposed Solution

After careful evaluation of different technologies that have been used for measuring soil moisture content, it appears that a capacitance based sensor would best meet the requirements set forth for this application, namely unattended long term reliability, low cost, and low energy. To better understand how this particular sensor type meets these specific characteristics, it is necessary to describe why it can be used to measure the water content of soil in the first place.

$$C = \frac{\varepsilon_0 \varepsilon_r A}{d}$$

Equation 3-1

A capacitor in its most basic form can be created using two metal plates (electrodes) separated by a dielectric with the capacitance value given by Equation 3-1. Assuming that the distance d between the electrodes and their area A does not change, the only variable component left is the dielectric constant ε_r. Most of the compounds that make up dry soil have a dielectric constant of 3-5 [8]. Soil is a very porous medium and a large portion of its volume is air which has a dielectric constant of 1. However, when the soil is wet, the volume occupied by air is now occupied by water which has a comparatively giant dielectric constant of 80. With this in mind, different levels of volumetric water content (VWC) will result in different dielectric constants [13]. Since the dielectric constant is dependent on the volumetric water content of the soil and all other parameters defining the sensor are fixed, measuring the capacitance of the probe can be used to directly determine the volumetric water content of the soil.

As described earlier, the sensing electrodes of a capacitance based sensor are completely encapsulated in an insulating material. Since water present in soil is far from pure and will contain dissolved salts and ions, a lack of insulation would allow for electrical shorting between the two electrodes of the sensor, which would prevent it from working capacitively. The insulation also plays a double role in preventing electrolysis and corrosion of the electrodes which would affect the accuracy of future measurements. With an insulating layer made of PVC or polyurethane, neither of which is non-reactive with water or other compounds commonly found in soil, the sensor can be left permanently installed for long periods of time with no requirement for maintenance cleanings, unlike the battery-style and resistive methods discussed previously that depend on exposed metal electrodes.

The various sensing technologies evaluated all had different strength and weaknesses. The more expensive technologies such as GPR and neutron probes provide much better measurements because they can average over larger volumes of soil and thus are not affected by localized variations in soil moisture or air pockets near the sensor. The low-cost battery and resistive style sensors on the other hand are heavily dependent on uniform constant contact with the soil and can only measure the moisture content of soil in very close proximity to the sensor. The style of capacitive sensor used in this project comes very close to the price point of other low-cost sensors especially with the manufacturing technique discussed further in this paper. While steps were taken to reduce the effects of air pockets and localized variations in soil moisture, like is often done with the resistive sensors such as the Watermark, the capacitive sensor is still susceptible to them. However, the low cost allows many more sensors to be installed in the field providing measurement

dependability comparable to the high cost sensors and the long-term corrosion resistance of the capacitive sensor gives it a significant advantage over other sensors that require periodic maintenance.

The need to have a sensing technology that takes very little energy for performing measurements is not a requirement that is immediately evident. However, when looking at this from the perspectives of maintenance free long-term installation and low cost, the need becomes much more obvious. To facilitate ease of installation and minimize potential interference with other work in an agricultural installation, the sensors cannot depend on wires being run for power or data transmission. This necessitates that the sensor has wireless capability for communication and be self-contained for its power needs (battery or solar powered). Technologies such as GPR and TDR could in theory be run off of a battery, but their power requirements are large enough that they could not run for the extended periods of time required without large (high cost) or often-replaced batteries. Capacitive sensing on the other hand is possible with very low amounts of energy actually being used for the measurement itself (determined in a subsequent section to be on the order of microjoules). The bulk of the energy is used for the actual wireless communication which would be the same amount regardless of the sensing technology used. This is because the sensor is just a very low value capacitor on the order of 50-200pF.

$$Q = CV$$

Equation 3-2

$$W = \frac{1}{2}\left(\frac{Q^2}{C}\right) = \frac{1}{2}CV^2$$

Equation 3-3

The final design of this system uses a 3V nominal supply that is used to charge the sensor to approximately 1.5V. Using Equation 3-2, this would result in the sensor having a charge of 150 picocoulombs. Converting this to energy (Equation 3-3) gives 112.5 picojoules, which can be considered a negligible amount of energy. The few microjoules used during measurement are primarily for running the low power optimized microcontroller and peripherals which would similarly also need to run for any other sensor type. The low energy requirements of a capacitance based sensor make it an ideal for a system that needs to operate for long periods of time on a low cost battery that need not be replaced (possibly for years).

3.1 Preliminary Sensor Designs

There are many geometries that can be used to build a capacitor. However, in this case there is an important design requirement that the chosen geometry must satisfy. Unlike standard capacitors where the dielectric material is an extremely thin layer sandwiched tightly between the electrodes, this capacitive sensor must allow the electric field lines between the electrodes to interact with the moisture from the surrounding soil, something a traditional parallel plate design does not allow for. There are several ways of achieving this, either by orienting the electrodes such that the field lines penetrate the soil or by drawing the moisture into some medium between the electrodes. To that end, pairs of insulated wires were formed into various geometries and evaluated.

$$C = \frac{\pi \varepsilon_r \varepsilon_0 l}{\ln\left(\frac{d}{2r} + \sqrt{\frac{d^2}{4r^2} - 1}\right)}$$

Equation 3-4

The capacitance between a pair of parallel wires can be coarsely estimated using Equation 3-4, where *d* is the separation between the wires, *r* is the wire radius, and *l* is the length of the segment. There is a caveat however as this equation assumes the separation between the wires is greater than their radius, as would be the case with twin-lead antenna cable for example. Many of the designs tested violated this assumption due to thin insulation and having the wires tightly twisted around one another (twisted pair). Since precise values were not the goal of these tests, but rather how different parameters relatively affect the performance, a precise calculation was unnecessary.

Wire Diameter & Type	Capacitance (pF/cm)		Factor
	Air	Water	
0.0159 in, 26AWG Cat5E - straight	0.5	1.8	3.6
0.0123 in, EC, twisted pair, straight	1.3	15.5	11.9
0.0320 in, EC, twisted pair, straight	2.0	27.6	13.8
0.0123 in, EC, twisted pair, coiled	0.9	14.0	15.6
0.0320 in, EC, twisted pair, coiled	1.3	27.9	21.5
0.0320 in, EC, cylindrical bifilar coil	3.2	18.2	5.7
0.0320 in, EC, flat bifilar coil	3.6	6.1	1.7

EC: enamel coated

Table 1: Summary of preliminary design tests using wire pairs

3.1.1 Twisted Pair Wire

Figure 3-1: 26 AWG twisted pair wire, PVC insulated

The initial experiment to test the concept of exploiting water's high dielectric constant involved the use of a single twisted pair of wires from a Cat5e network cable. The ends of the wires were insulated to prevent shorting between the electrodes when immersed in water. Though crude, this turned out to be an easy and effective test of the theory. The wire used was 26AWG (0.40 mm) copper insulated with 0.245 mm of PVC sheathing, providing 0.490 mm of separation between the two electrodes [Figure 3-1]. In air, the 95cm length of wire provided a capacitance of roughly 48pF which calculates to roughly 0.5pF/cm of wire [Table 1]. Despite the thickness of the insulating materials, when a 17cm segment of the wire was immersed in water, the total measured capacitance rose to around 70pF and promptly returned to 48pF once removed from the water. A simple calculation to remove the effects of the segment of wire not immersed in the water shows that in air the 17cm segment had a capacitance of around 8.5pF while in water it rose to 30.5pF. The experiment shows that despite its non-optimized design, there was a clearly measurable increase of 3.6 times in the measured capacitance between a dry and wet "probe".

Figure 3-2: 0.0320 in. diameter, coiled, twisted pair wire, enamel coated

Figure 3-3: 0.0132 in. diameter, coiled, twisted pair wire, enamel coated

Assuming Equation 3-4 holds somewhat true, a decrease in the distance between the electrodes should increase the effect of water's dielectric constant on the capacitance value. To minimize the distance between the two electrodes, a pair of enamel coated wires was used such as is commonly found in transformer wiring. Two separate 27 cm length sensor probes were made using 0.0320 and 0.0123 inch diameter wire formed into a twisted pair as in the preceding Cat5e wire experiment. The enamel coating was determined to be approximately 0.0005 inch (0.0127 mm) in thickness by measuring the outside diameter of the coated wire compared to the same wire that had the enamel chemically dissolved away. The ends of the wires were insulated just as before to prevent electrical shorting by the surrounding water.

The first set of experiments measured the capacitance value of a straight segment of each twisted pair in air versus immersed in water. By bringing the electrodes closer together (by reducing insulation thickness roughly 95%), the capacitance has increased by a factor of 3.3 to 3.8 with the smaller and larger diameter enamel coated wire respectively. The larger diameter wire had a slightly higher capacitance value which follows Equation 3-4, indicating that as the cross sectional area of the electrode increases, so does the capacitance.

The second set of experiments were an attempt to test the effects of forming the enamel coated wire twisted pair into a coil in an effort to reduce the physical dimensions occupied by the sensor. Each twisted pair was coiled around an 8mm diameter mandrel and stretched to form 10mm spacing between the segments [Figure 3-2][Figure 3-3]. Coiling the twisted pair wire had the unforeseen effect of introducing intermittent air gaps (< 1 mm) between the two wires of the twisted pair. The effect is seen as a decrease in

capacitance values for both sizes of enamel coated wire when in air. This correlates with Equation 3-4 which states that an increase in the spacing between electrodes reduces the effective capacitance between them. Interestingly however, when the coiled twisted pairs were immersed in water, their capacitance was greater than that of the same length segments in a straight configuration. This is likely due to the fact that despite the minimal increase in separation between the wires, the effect of the water's dielectric constant was much higher as it now filled the space between the wires where electric field lines are strongest, a space that was previously occupied primarily by the enamel coating which has a low dielectric constant. These results show that a small air gap between the two electrodes of the sensor is beneficial to allow the presence of water where electric field lines are strongest, thereby increasing the sensors sensitivity.

3.1.2 Bifilar Coil

To maximize measurable capacitance while at the same time reducing the size of the sensing element, a pair of enamel coated wires was hand wound into a cylindrical bifilar coil configuration forming a hollow cylinder [Figure 3-4]. An interesting side effect of this design is that nearly every segment of one electrode is now between two segments of the other, and vice versa. This configuration allows for twice the coupling between the electrodes and should double the effective capacitance of the probe, making measurement easier and less susceptible to noise due to the capacitance values being much larger. For ease of manufacturing the wires were tightly spaced, leaving no room for water between the electrodes, hence the lower "in water" measurements when compared to the looser twisted pair coils.

Figure 3-4: Cylindrical bifilar coil, enamel coated wire

One significant problem with the cylindrical configuration of the coil is the hollow "core". If the core is left as an air pocket, it could potentially fill with water that does not drain out easily, which would shift subsequent capacitance measurements, throwing off the moisture reading of the surrounding soil. On the other hand, if the core is filled with a porous hydrophilic media, such as gypsum, the response time of the sensor could be excessively long since only the small ends of the cylinder (assuming a relatively long cylinder) are exposed to the surrounding soil. Having a solid core made of a water impermeable material resolves the problems of the preceding cases, but prevents water from interacting with the "inside wall" of the cylinder, effectively cutting the capacitance in half and defeating the previously discussed doubling.

Figure 3-5: Flat bifilar coil, enamel coated wire

To overcome the problems of the solenoid coil design's hollow core, the same length of wire was formed into a flat bifilar coil [Figure 3-5]. Just as in the cylindrical design, each electrode is between two segments of the other electrode thereby doubling the effective capacitance over the length of the electrodes. Also, for similar ease of manufacturing reasons, the tight wire spacing left no room for water between the electrodes. The flat coil results in a very small and slim sensor package which should have very similar capacitance characteristics to the cylindrical coil design. Unfortunately, a significant amount of epoxy was needed to keep the shape of the coil static which added undesirable thickness to the insulation of the electrodes, thereby lowering the "in water" measurements significantly. Despite this setback, the similarity in capacitance between flat and cylindrical coil design for "in air" measurements attests to independence with regard to coil geometry so long as electrode length and spacing are preserved as well as having each electrode be between two segments of the other electrode.

Despite the benefits and performance of the bifilar coil design, it is difficult to manufacture by hand and would require customized tooling for an automated process. To mitigate this, a design based around similar principles was implemented using an already existing low cost manufacturing process and form factor.

3.2 Final Sensor Design

Using a printed circuit board (PCB) for the sensor electrodes gives multiple advantages over the previously discussed enamel coated wire construction. The biggest benefit is that for a relatively low cost, the sensors can be manufactured to a high degree of consistency resulting in minimal variation between individual units. Additionally, the

planar design of the sensor reduces overall size resulting in a smaller package for a given capacitance and does not have the issues of the hollow center core present in the cylindrical wire design.

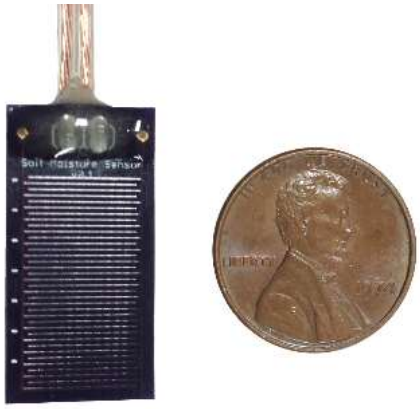

Figure 3-6: Sensor probe

Figure 3-7: Sensor probe PCB design

The sensor was designed with two electrodes per side of the PCB to maximize capacitance. The electrodes were shaped as interdigital combs containing 36 pairs of 10mm long fingers on each side of the PCB resulting in 700mm of effective capacitive length [Figure 3-7]. This would roughly translate to 70cm of twin lead wire in a coil design. Using the inverse relationship between electrode spacing and effective capacitance, the traces were spaced 0.010 inch (0.254 mm) apart to minimize separation while still being reasonably above the 0.006 inch minimum tolerance allowed by the board manufacturer. Maximizing the effective capacitance of the sensing portion of the probe makes it the dominant factor in measurements. The trace width was chosen as 0.010 inch. The number of interdigitated fingers is the maximum number that could fit within the desired dimensions of the sensor being 0.5 x 1.0 inches with space left for the wire connection and associated insulation. To minimize capacitance between electrodes on opposite sides of the

PCB, the copper traces forming an electrode on one side of the board are positioned directly above the copper traces forming the same electrode on the other side of the board.

An approximation of the total capacitance for the PCB probe design can be calculated using Equation 3-5 where ε_r is the dielectric constant of the FR4 substrate, l is the length of each finger, A_1 and A_2 are the capacitances per unit length of the, and N is the total number of fingers [14, p. 230].

$$C = (\epsilon_r + 1)l\big((N-3)A_1 + A_2\big)$$

Equation 3-5

Since spacing between the fingers is equal to the finger width and the substrate has a finite thickness, A_1 and A_2 can be approximated as follows [14, p. 232]:

$$A_1 = 4.409 \tanh\left[0.55\left(\frac{h}{W}\right)^{0.45}\right] \times 10^{-6}$$

Equation 3-6

$$A_2 = 9.92 \tanh\left[0.52\left(\frac{h}{W}\right)^{0.5}\right] \times 10^{-6}$$

Equation 3-7

To account for the different dielectric constants of the probe's surroundings, Equation 3-5 is modified by replacing ε_r with k_1 for the dielectric constant of the substrate (FR4) and substituting 1 with k_2 for the dielectric constant of the probe's surroundings since it is no longer necessarily air. Additionally, the whole capacitance is multiplied by a factor of 2 because there are two sides to the probe with the same design. The new approximation becomes:

$$C = 2(k_1 + k_2)l\big((N-3)A_1 + A_2\big)$$

Equation 3-8

The probe design used the following parameters: dielectric constant of substrate k_1 = 4.5, thickness of the FR4 substrate h = 1.6 mm, width of fingers and spacing W = 0.010 in (0.254 mm), length of fingers l = 0.380 in (9.652 mm), number of fingers per face of probe N = 36. Solving Equation 3-8 for a dry (k_2 = 1) and a fully water immersed (k_2 = 80) probe yields a capacitance of approximately 14 pF and 216 pF respectively.

These values are only idealized approximations that, although close to empirical measured values, do not take into account two primary sources of discrepancy. The first is the additional capacitance from the wire leads between the TPIP-K and the probe. The other is the insulating conformal coating of the probe PCB which varies in thickness (approximately 50-75 μm) and has a dielectric constant of approximately 4. This results in the total capacitance of the probe being slightly higher when dry (total effective dielectric constant higher than air) and lower when wet (total effective dielectric constant lower than water).

Early testing revealed that despite the FR4 material of the board having low water permeability, enough water entered the board material to have a noticeable increase of the nominal capacitance of the sensor. This necessitated baking the sensor boards at 100C for a minimum of 1 hour to remove moisture in the board material followed by sealing of the board edges with urethane conformal coating to prevent moisture from reentering the board material. Further inspection under a microscope revealed that while the solder mask, intended to keep the traces insulated from the water, looked uniform to the naked eye, it actually contained tiny pits that left some of the copper exposed to the water allowing for electrolysis and leakage currents (Figure 3-8).

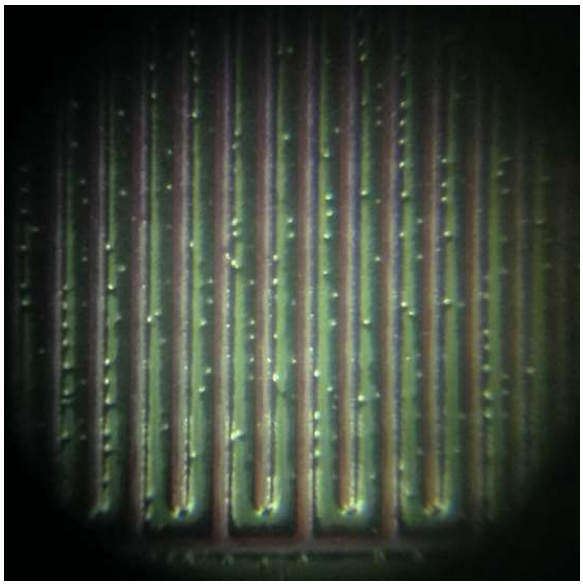

Figure 3-8:Microscopic pits in PCB solder mask

The application of an additional layer of conformal coating fills these imperfections in the solder mask without affecting the capacitance of the sensor board noticeably. Subsequent tests showed that prolonged water immersion did not result in the same increase in nominal capacitance as with the unsealed boards.

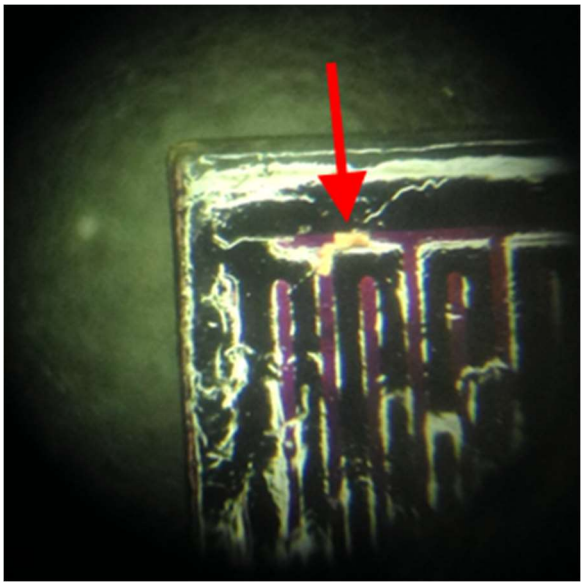

Figure 3-9: PCB trace damage

During in-soil experiments, it was discovered that the solder mask coating of the bare sensor board developed hairline scratches and sometimes chipped enough to expose tiny sections of the copper traces (Figure 3-9). This resulted in leakage currents which, while not enough to short out the sensor entirely, caused erroneously high and erratic measurements to occur. Resistance measurements of a scratched sensor immersed in tap water showed values starting in the 6-7MΩ range quickly rising to >12MΩ. The rise in resistance was likely due to electrolysis of the copper resulting from the current used by the multimeter to measure the resistance. Baking the sensor and coating the electrode surface with urethane conformal coating, as in the edge sealing procedure, restored the sensor's resistance to >50MΩ (the multimeter used could not measure higher values). However, the additional layers of conformal coating on top of the prior applied layer increased the thickness of the insulation over the electrode traces reducing the maximum in-water capacitance to ~150pF. A similar effect was observed during construction of the flat bifilar coil due to excess epoxy. This sort of damage to the sensor board, despite careful installation in the soil, necessitates a case of some sort that protects the sensor while at the same time does not compromise its ability to measure moisture levels in the surrounding soil.

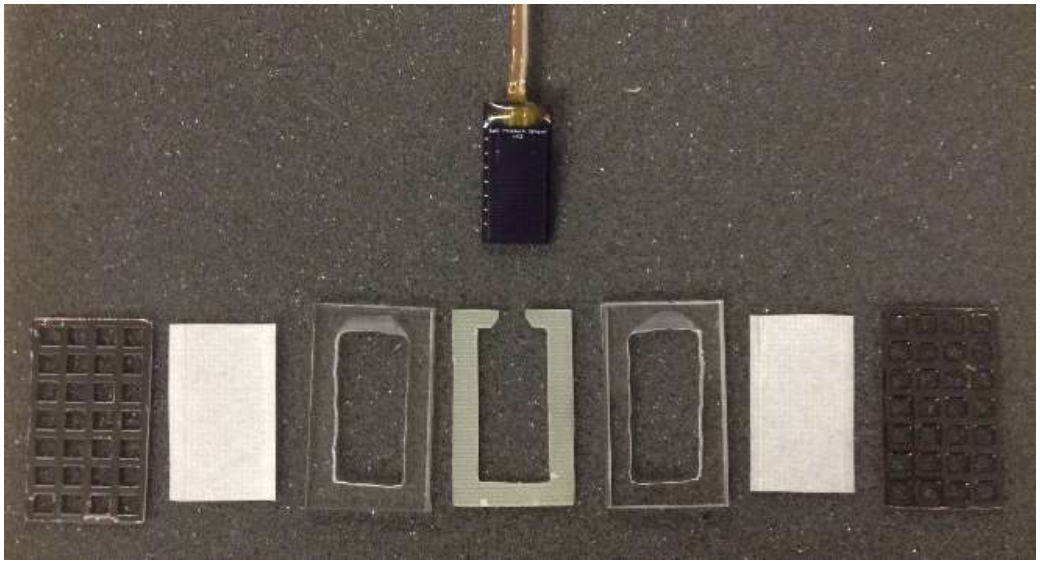

Figure 3-10: Complete probe, individual component view

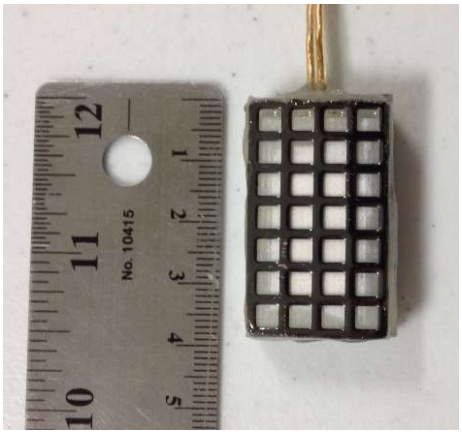

Figure 3-11: Completed probe, front view

Figure 3-12: Completed probe, side view

To protect the sensor board from being scratched by rocks and other debris normally present in soil, it must be encased in a protective housing. Unfortunately, for the sensor to function, it must be in close contact with the moisture present in the surrounding soil. To solve these two problems, the sensor board is housed in a sandwich like package [Figure 3-10]. Just as the two sides of the sensor board are symmetrical, so are the two

halves of the complete housing package. The sensor board is fitted in a plastic frame containing a cutout equal to the dimensions of the sensor board itself [Figure]. Then a plastic frame with a cutout matching the vital area of the sensor is placed on top [Figure]. This open space is filled with diatomaceous earth [Figure] which was chosen because its hygroscopic properties will draw water from the surrounding soil and the fine uniform particles will fill the gaps between the traces on the PCB ensuring moisture is kept close to the sensor board. To prevent the ultra-fine diatomaceous earth from escaping, it is subsequently covered with a 50 micron polypropylene filter material (approximately 0.010 in. thick) commonly used in household water filters, so it will let water through but prevent particles from flowing into or out of the sensor. Finally, a coarse metal mesh is placed over the filter material to prevent damage from large sharp objects [Figure 3-11]. All layers are glued together with water resistant epoxy around the edges leaving only the filter material and diatomaceous earth between the sensor board and the soil. This design is admittedly not ideal for actual real-world use, primarily due to the shape, but it is sufficient to test the characteristics of the sensor design.

3.3 Wireless Backhaul

Figure 3-13: TPIP-K wireless tag

The TPIP-K wireless tag, developed at WINLAB, was selected as the controller and wireless backhaul not only for being readily available but also for its ultra-low energy

characteristics and low cost. Based around a TI MSP430G2553 microcontroller paired with a TI CC1150 RF transmitter, the TPIP-K is specifically designed for remote sensing applications where size, cost, and battery life are of critical concern. The firmware running on the TPIP-K's MSP430 is optimized for low energy usage, having the tag take measurements and transmit data at specified intervals and entering an ultra-low power deep sleep mode in between to conserve battery life. The transmit-only design significantly reduces energy consumption because no bi-directional communication takes place, allowing the radio chipset to only be active for extremely brief periods when data needs to be reported, unlike other technologies such as Bluetooth and Wi-Fi. With a typical expected lifespan of eight to ten years when running off a single CR2032 coin cell battery and a low production cost per unit, the TPIP-K is the ideal choice for a soil moisture monitoring system. The zero-maintenance nature of the TPIP-K lends itself to years of use in permanent installations.

3.4 Sensor Probe Interface

Since most of the actual components necessary for measurement are built into the MSP430 on the TPIP-K, a simple circuit is all that is necessary to interface with the moisture probe. Each of the software implemented measurement methods evaluated require a slightly different hardware interface, but they generally fall into one of two groups. The first is a direct connection which requires only a single I/O pin (used for sensing/measurement) and no additional components between the probe and the TPIP-K. One electrode of the sensor probe is connected to the I/O pin, the other is connected to ground. The second group of methods makes use of an additional I/O pin (used for

controlling probe charging/discharging) and a resistor of known value which is used to control the charge/discharge rate of the probe [Figure 3-15]. Having the resistor R be of a known value makes calculation of the measured capacitance trivial. The functionality of the pins, determined by the software method in use, is described in the following sections.

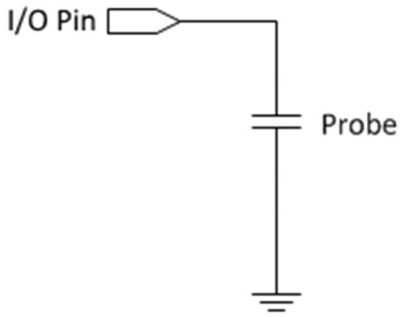

Figure 3-14: Single pin sensor probe schematic

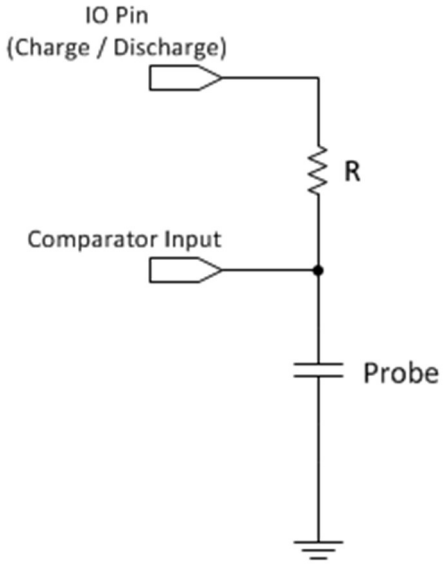

Figure 3-15: Two pin sensor probe schematic

3.5 Software

The principle on which this sensor design operates, a variable capacitor whose dielectric is the moisture in the surrounding soil, lends itself to a very simple measurement:

the capacitance of the sensor at some soil moisture level. If the capacitance is known, then the corresponding water content of the soil can be determined via the linear relationship proportional to the volumetric water content of the soil.

3.5.1 Methods of Measurement

The two circuit designs previously discussed combined with the functionally reconfigurable pins of the TPIP's MSP430 microcontroller allowed for the evaluation of several different capacitance measurement techniques via software. The ability of the MSP430 to near arbitrarily connect physical pins to its internal modules allows for these complex configurations to be implemented with little to no changes in external circuitry. The methods evaluated could be effectively divided into two distinct categories. The first involves direct measurement of the capacitance by determining the time it takes to charge or discharge the probe between two known voltages. The second category uses the probe as part of an oscillator circuit whose measured frequency depends on the probe capacitance.

3.5.1.1 Input Pin Threshold

Of the techniques described, using the input pin threshold to trigger a timer capture event is the simplest. The time it takes to charge or discharge the probe to a specific voltage is measured which can then be translated into a capacitance and subsequently correlated with a soil moisture content level. The external circuit is setup as in Figure 3-15. The "sensing" pin is configured as a high impedance input which triggers a capture of the timer value once the voltage across the probe, being charged (or discharged) through resistor R by the "charging" pin, exceeds (or falls below) the input threshold voltage of the pin.

The major problem with this method is the dependence on the threshold voltage of the pin to determine when the probe voltage reaches a desired point. The MSP430 uses Schmitt trigger inputs which are designed with hysteresis built in to improve digital performance. This means that there is a different threshold voltage for rising and falling slope signals. Fortunately, the direction of the slope is known because it is determined by whether the probe is being charged to Vcc or discharged to ground. Unfortunately, because the Schmitt trigger inputs are designed for digital signals, the manufacturing tolerances allow for as much as 0.9V of variation in the threshold values between different pins [15, p. 24]. The result is that the charge/discharge time measured is to an unknown voltage, which makes determination of the capacitance value impossible. The threshold voltages can be determined empirically and calibrated for; however, the CMOS gates on the chip are temperature sensitive enough that the threshold values can change 2 - 4mV/°C [16] which is significant over the expected temperature range of 5 - 50°C. Without a known threshold voltage, this method for determining the capacitance allows for too much error in the measurement.

3.5.1.2 Pin Oscillator

This technique falls into the second category of measurements described earlier. By using the probe as part of an oscillator circuit whose resonant frequency depends on the capacitance of the probe, a measurement of the resulting frequency can be used to determine the capacitance of the probe. There are several ways to implement this oscillator, many of which are described in an application report from Texas Instruments regarding

capacitive touch [17]. The oscillator based implementations described in that report require the use of external resistor networks to provide reference voltages.

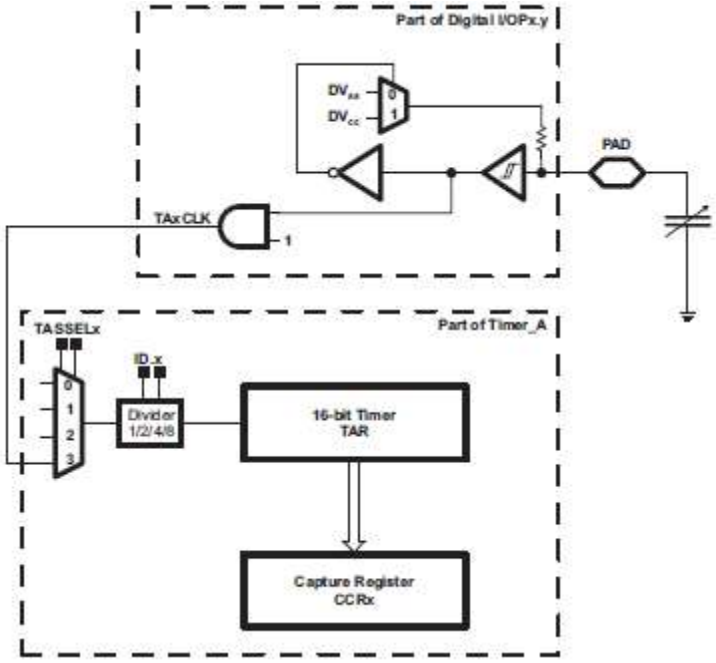

Figure 3-16: Pin oscillator function [18, p. 330]

To avoid the need for external components, the MSP430 used on the TPIP-K has a "capacitive sensing" capability on some of the pins. This is normally used for implementing capacitive touch controls such as buttons but can be readily adapted for use in measuring the capacitance of the soil moisture probe. The capacitive sensing function effectively works by enabling both the input and output functions of a pin and uses an inversion of the input signal as the control for the output module [Figure 3-16]. The result is the pin voltage oscillates between ground and Vcc as the pin proceeds to pull itself low when its input reads high and pulls itself high when its input reads low. When the probe is connected as in Figure 3-16, the capacitance of the probe combined with the internal pullup/pulldown resistor form an RC network which determines the oscillation frequency

of the pin. As the capacitance of the probe rises with increases in moisture level, the resonant frequency of this RC oscillator will decrease. Using two of the built-in timer modules, one using the oscillating pin input value as a clock source and the other using a separate constant frequency clock source, the number of oscillations in a predetermined finite amount of time can be counted. The number of oscillations counted can be mapped to particular soil moisture levels through calibration. Unfortunately, the number of oscillations counted is not linearly dependent on the capacitance [18, p. 331]. Additionally, like the input pin threshold method, the pin oscillator suffers from a highly unknown variable. This time it is the pullup/pulldown resistor of the particular pin which is specified as being 35kΩ nominal but can range from 20k-50kΩ due to manufacturing tolerances [15, p. 24].

3.5.1.3 RC Charge

This method follows the technique used to solve the classic electrical engineering problem of determining an unknown capacitance by measuring the time it takes to charge the capacitor to a specific voltage. Using the interface circuit in Figure 3-15 between the TPIP-K and the moisture probe, the capacitance value of the probe can be determined by using the Timer_A module to measure the number of clock cycles it takes to charge the probe to a known voltage. This threshold is monitored using the SENSE pin which is connected to one of the Comparator inputs. Assuming the capacitor is initially at 0V, for a particular time *t*, the capacitor will charge to a specific voltage according to the following equation.

$$V_{final} = V_{supply}\left(1 - e^{\frac{-t}{RC}}\right)$$

Equation 3-9

The value of V_{final} has an exponential dependence on the time t it takes to reach that value. Since the real unknown in this case is the capacitance value C, the formula can be rewritten as follows.

$$C = \frac{t}{R \ln\left(\frac{V_{final}}{V_{supply}}\right)}$$

Equation 3-10

The dependence between C and t is strictly linear assuming all other parameters remain known and unchanged. The value of V_{supply} in this case would be the value of V_{cc} supply line on the TPIP-K and can be measured using the onboard ADC. However, the ADC cannot be used to determine the point in time when the voltage across the probe reaches V_{final} because it takes time a relatively long time to perform an ADC measurement (> 0.3 µs on the TPIP-K) when compared to the rate of change in voltage across the probe (30-120µs to reach ½ V_{cc}) and readings would have to be taken and compared continuously in order to catch the threshold voltage. Additionally, the internal reference voltages can vary as mush as ±6% [15, p. 38]. To solve this issue, the onboard Comparator_A+ module can be used which triggers in hardware when a sensed input voltage is greater than a given reference voltage with a response time of as little as 120ns [15, p. 35]. Another benefit of using Compoarator_A+ is that the optional internal reference voltages are not fixed value like they are for the ADC, but linearly dependent on V_{cc}. This eliminates the need to know what V_{cc} is. Selecting ½ V_{cc} as the comparator's reference voltage reduces Equation 3-10 significantly:

$$C = \frac{t}{R \ln\left(\frac{1}{2}\right)}$$

Equation 3-11

Now the calculation of the sensor probe's capacitance is very simple. All that is needed is a known resistor R, which is used to determine the charging rate of the capacitive probe, and the time it takes to charge the probe from 0V to ½ V_{cc}. To maximize accuracy and repeatability, rather than implementing the measurement algorithm in software, the MSP430's internal hardware interconnects between Timer_A, Comparator_A+, and GPIO control modules were used to create a hardware based "stop watch".

After extensive testing and optimization, the RC charge algorithm was implemented as follows [Figure 3-17: Measurement algorithm in softwareFigure 3-17]. Original source code implementation is provided in the appendices of this paper. The probe is first discharged through the CHARGE and SENSE pins concurrently by pulling them low (the SENSE pin effectively shorts to ground). This gives a reliable starting point of 0V across the probe in under 20µs even if the probe is charged to 3V (Vcc). The SENSE pin is then reconfigured as an input to the Comparator_A+ module, while the CHARGE pin remains pulled low, and Timer_A is preset at a high count value. Timer_A has the ability to change the state of an output pin (in this case the CHARGE pin) from low to high (or vice versa) when it "rolls over" back to zero upon counting up to its maximum possible value. By presetting the timer count, this hardware functionality can be taken advantage of without having to count through the full range of 65535 which would waste 10ms with a Timer_A clock source of 6.5MHz. Additionally, the use of Timer_A to control when the CHARGE pin is pulled high effectively synchronizes the timer to the actual start of

charging. Once the MSP430 is properly configured, Timer_A is started and the CPU is put into low power mode to conserve energy. As Timer_A reaches its maximum possible count value and starts counting from zero, the CHARGE pin is pulled high and begins to charge the probe through the known resistor R [Figure 3-15]. When Comparator_A+ senses that the voltage across the probe has reached ½ Vcc, it triggers a Timer_A capture event via a previously configured internal hardware connection. By triggering the capture in hardware, the near exact time that the probe voltage crosses ½ Vcc can be recorded. The capture event also triggers an interrupt, waking up the CPU from low power mode and stopping the modules from triggering further events erroneously. The timer count value captured in Timer_A is added to a running total for later averaging. A check is then performed to see if the desired number of measurements has been completed. If not, the process repeats, otherwise the sum total of measurements is averaged and further data processing is conducted, such as applying calibration and scaling functions to remove known offsets and translate the measured values to a desired scale. The finalized value is then passed on for transmission.

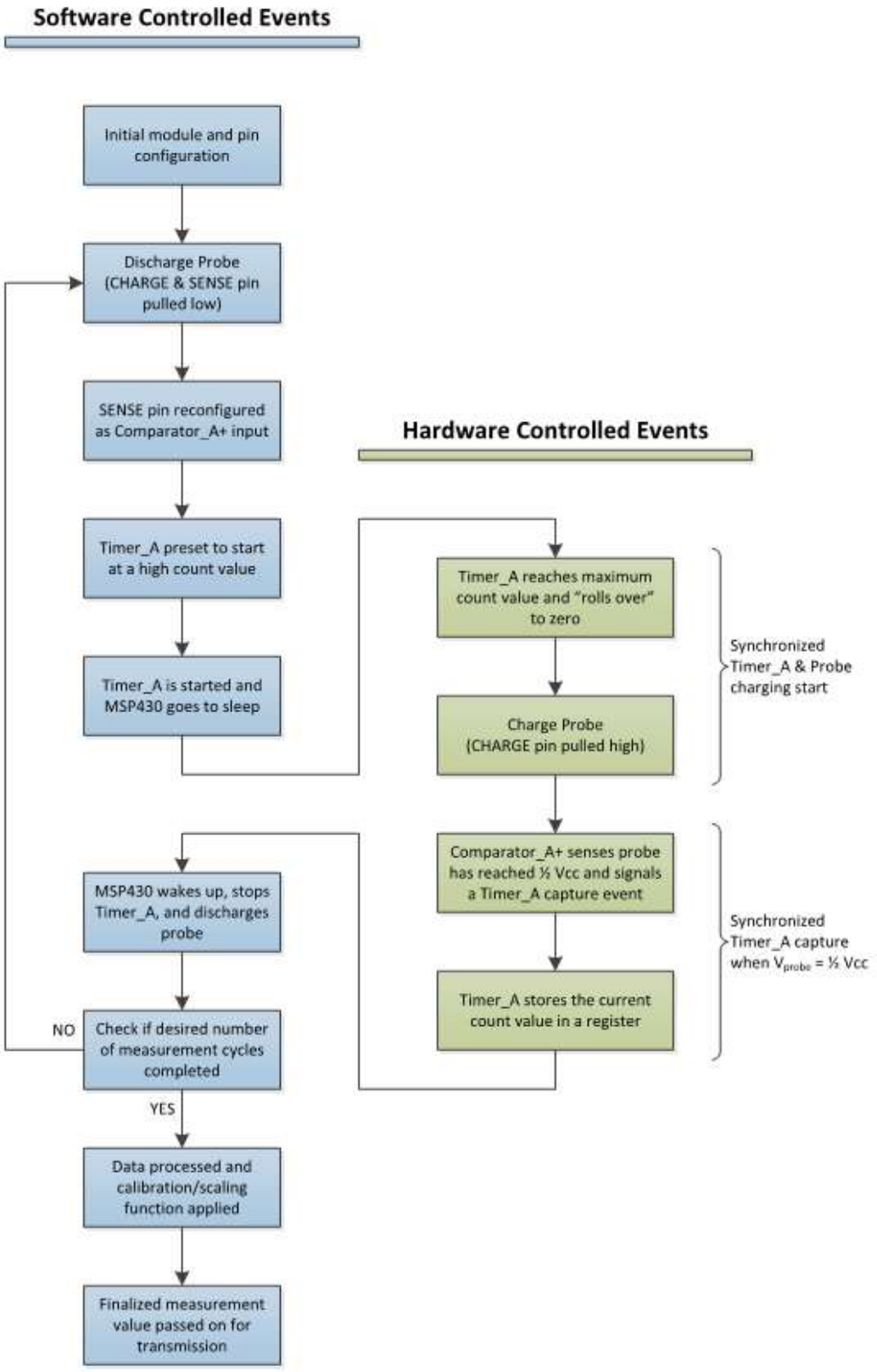

Figure 3-17: Measurement algorithm in software

The idea of determining a capacitance value by measuring the time it takes a capacitor to charge up to a known voltage level is nothing new. This method is readily implementable in software. However, it can introduce errors and offsets in determining the precise start and end times of the charging because software takes time for the CPU to execute and many of the steps typically cannot occur at the same time because they require different commands to be executed. Given that the MSP430 and most other microcontrollers are not parallel architectures, there would be an inherent delay between the start of the timer and the start of charging the probe for example. This delay could technically be measured for a known set of code, but it is not trivial and must be redone any time related code is modified. The novel approach described here takes advantage of the hardware modules of the MSP430 and their interconnections to transfer the entire measurement process to a purely hardware-controlled system with timing synchronization on the order of less than one clock cycle. This automatically removes any software-based timing offsets and is code independent.

Minor changes to the implementation gave measurable improvements in accuracy. The comparator on the MSP430 has a feature (CAEX) allowing the internal swapping of the comparator inputs to account for some of the voltage offsets present in the module. By taking 2n measurements and enabling CAEX on exactly half of them, half the measurements will trigger just below the reference voltage (Vcc/2 in this case) and half just above. This under/overshoot offset will be a relatively fixed value which disappears when 2n such measurements are averaged. Experimental testing resulted in 0.5 to 1% difference between pairs of measurements where CAEX was and was not enabled. Averaging multiple measurements also results in a significant reduction in the effect of

uncorrelated noise. In an ideal case, for *n* measurements, the noise is reduced by a factor of 1/n. Simply taking one pair of measurements to make use of the CAEX functionality, not only removes the offset error of the comparator but reduces the noise level in the measurement to 50% that of a single measurement. By taking two pairs of such measurements, the effect of noise is reduced to half of its original level for only double the energy expenditure. Taking more pairs of measurements tends to follow the law of diminishing returns with regard to energy. Extensive experimentation, discussed in greater detail in a subsequent section, determined that averaging 4 individual measurements resulted in a stable and relatively noise free measurement value while keeping energy use to a reasonable level.

3.5.1.4 RC Charge and Discharge

This method is very similar to RC charge in implementation, but differentiates itself functionally in two significant ways. In addition to slowly charging the capacitive probe through a resistor R, this method also slowly discharges the probe through that same resistor instead of quickly via shorting to ground. This permits the additional measurement of the discharge time to a known voltage, which may remove some additional small offset induced errors. The second difference is that, instead of waiting for the probe to fully charge to Vcc or discharge to ground, the charge/discharge pin is switched preemptively and does not allow the probe voltage to reach either extreme. This causes the voltage across the probe to oscillate up and down, with its median value settling naturally toward Vcc/2 within a few oscillations. A single measurement is taken by determining the time between the moment the charge/discharge pin is flipped and the point at which the probe voltage

crosses Vcc/2 (determined using the comparator). By shortening the time spent waiting between flips of the charge/discharge pin, the oscillations occur more frequently and many more measurements can be taken in the same time period as one measurement using the RC charge method. By averaging over many measurements, errors due to noise are reduced and some interpolation occurs to potentially improve measurement resolution. However, there is an important caveat with shortening this time between flips to increase the number of measurements. By shortening the time period, the starting point of measurement is moved closer to the point when the probe voltage crosses Vcc/2. Because the difference between the starting point and crossing point depends on the capacitance being measured, care must be taken during implementation to ensure that the starting point allows enough time for the probe voltage to cross Vcc/2 across the range of expected capacitance values. Otherwise, the probe voltage will never cross Vcc/2 and no measurement will take place. Due to the added complexity of this method and the reliability of the prior charge-only method, it was not pursued beyond a conceptual stage.

3.5.2 Timing Source

Determining the relative capacitance of the probe depends heavily on having a reliable and stable timing source regardless of which measurement method is used. On the TPIP-K, there is a crystal oscillator which is used by the frequency synthesizer in the radio chip for signal transmission and reception. This crystal has a frequency of 26MHz and is specified to a tolerance of ±10 ppm over a temperature range of -10 to 60°C [19], which is more than enough for agricultural applications. The radio chip is capable of outputting a clock signal to the MSP430 microcontroller based on the crystal frequency and a selectable

divider. On the MSP430, this clock signal can be used as the clock source for the Timer_A module.

On paper, the radio chip's crystal clearly outperforms the internal DCO of the MSP430 in terms of stability. To verify this empirically, data was taken at different temperatures and battery voltages with fixed capacitors substituting for the probe capacitance. The capacitors used were chosen specifically for their excellent temperature stability (TDK FK series with ±30ppm/°C over a temperature range of –55 to 125°C [20]).

	% increase in measurement		
Clock Source	47pF	150pF	180pF
Internal DCO	0.4	0.4	0.3
Radio Crystal	0.8	0.4	0.4

Table 2: Effect of temperature when varied 20°C to 50°C, battery voltage stable at 3.0V

	% decrease in measurement		
Clock Source	47pF	150pF	180pF
Internal DCO	1.3	0.7	0.9
Radio Crystal	1.2	0.5	0.5

Table 3: Effect of battery voltage when varied 2.7V to 3.0V, temperature stable at 20°C

Using the radio's clock output, while providing a very stable clock source, does not come without consequences. By putting the MSP430 microcontroller into LPM3 mode, where the digitally controlled oscillator (DCO) is turned off, the sleep current is lowered to less than 1 uA. However, the radio chip must be turned on to provide the necessary clock source to the Timer_A module. This can increase the total current draw to over 3mA depending on the selected clock output frequency from the radio. Data was taken across the possible range of selectable frequencies from 1.625 to 26 MHz [Figure 4-3]. Despite the Timer_A specification stating not to exceed the clock frequency of the microcontroller,

the module appeared to work just fine even at the full 26MHz clock supplied by the radio which is 62.5% above the rated internal clock of the MSP430G2553 and over 4.3 times the 6MHz clock used on the TPIP-K. While impressive, the resolution provided at 26MHz is completely unnecessary in this application and the current draw significantly exceeds the ability of the battery to supply safely. By staying within the bounds of useful resolution and reasonable energy usage, the current draw during measurement can be estimated (±2%) using a linear approximation when deciding on the optimal frequency.

3.5.3 Calibration and Scaling Using Bilinear Interpolation

Naturally, no two sensor/PIP tag combinations will produce the same raw value reading despite being in the same environmental conditions. There are slight differences in each individual sensors as well as each PIP tag resulting from manufacturing tolerances both at the board and component levels. In addition, the voltage of the tag's battery was observed to affect the measurements slightly despite not being a factor in the expected mathematical calculation [Equation 3-11]. This is likely due to other components (such as the MSP430's internal timer and comparator circuits) having a supply voltage dependence which shows up as a variation in the moisture measurement readings. As a result, a calibration method must be applied to reduce the effects of these variations on the measurements.

A standard CR2032 lithium coil-cell is typically rated for a nominal voltage of 3.0V [21] though they often read around 3.3V with no load. At low current draw (<1mA), the battery remains close to 3V for much of its lifespan. When the battery voltage crosses approximately 2.6V, the amount of usable energy capacity remaining is comparably

minimal and, for this use case, considered near end-of-life. Battery dependency calibrations were conducted at 3.0V and 2.6V with the intent to improve measurement stability over the usable lifespan of the battery.

Raw measurements were taken at 3.0V and 2.6V supply voltages (representing "new" and "end of life" batteries) using high precision fixed value capacitors in place of the sensor probe to represent low and high measurement values. Capacitance values of 47pF and 180pF were used as they were closest standard capacitor values to the approximate sensor probe capacitance range of 50-200pF when dry and wet respectively.

A significant improvement occurred with the application of a bilinear interpolation based calibration method. Four individual data points are used where each represents the raw measurement value at a known moisture level and a known battery voltage. One point pair corresponds to a high value (e.g. 80% field capacity) on the desired measurement scale while the other will correspond to a low value (e.g. 20% field capacity). For each point pair, one point will correspond to a relatively low battery voltage (e.g. 2.6V) while the other will correspond to a relatively high battery voltage (e.g. 3.0V). Using these four data points, the raw measurement value can have its offsets and voltage dependencies compensated for while at the same time being translated to a desired measurement scale. When applying this calibration method, first two points are calculated which represent the expected measurement values corresponding to the low and high scale values at the measured nominal battery voltage [Figure 3-18]. Then the relative position of the raw measurement to these expected measurements is calculated and the corresponding scale value is determined through interpolation.

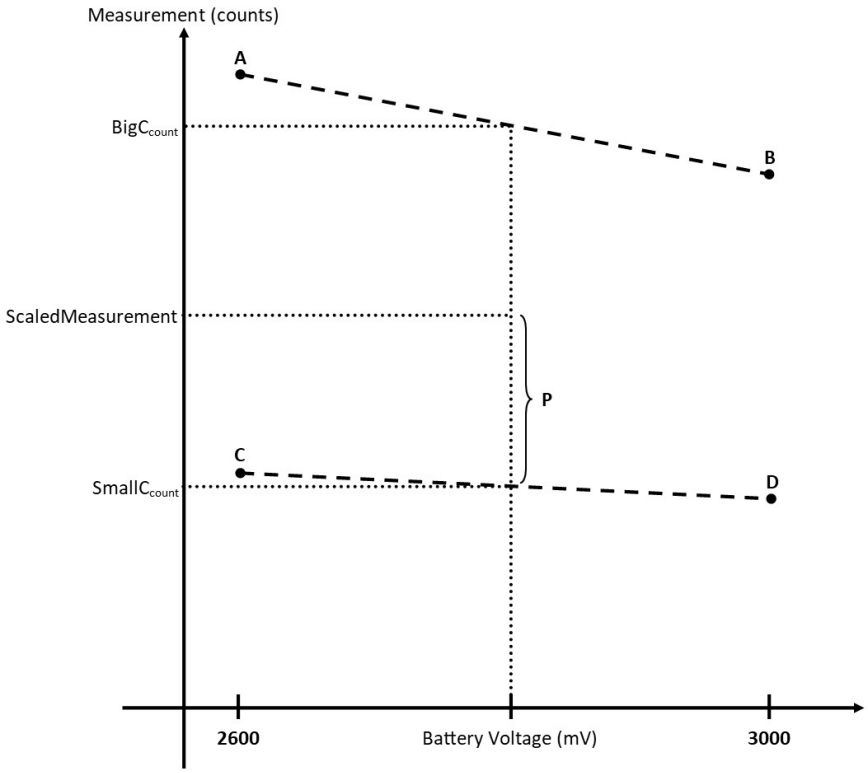

Figure 3-18: Calibration and scaling using bilinear interpolation

$$BigC_{slope}{}^{-1} = \frac{B_{voltage} - A_{voltage}}{A_{count} - B_{count}}$$

$$SmallC_{slope}{}^{-1} = \frac{D_{voltage} - C_{voltage}}{C_{count} - D_{count}}$$

$$BigC_{intercept} = A_{count} + \frac{A_{voltage}}{BigC_{slope}{}^{-1}}$$

$$SmallC_{intercept} = C_{count} + \frac{C_{voltage}}{SmallC_{slope}{}^{-1}}$$

$$BigC_{count} = BigC_{intercept} - \frac{Battery_{voltage}}{BigC_{slope}{}^{-1}}$$

$$SmallC_{count} = SmallC_{intercept} - \frac{Battery_{voltage}}{SmallC_{slope}{}^{-1}}$$

$$P = \frac{BigC_{count} - SmallC_{count}}{measurement_{count} - SmallC_{count}}$$

$$Measurement_{scaled} = SmallC_{scaled} + \frac{BigC_{scaled} - SmallC_{scaled}}{P}$$

Equation 3-12

The 16-bit architecture of the MSP430 microcontroller used on the PIP tags resulted in several limitations on the two-by-two point calibration algorithm implementation. Using regular integer variables limits every calculation step to a maximum value of +/- 32,767. To solve this, unsigned integers were used which increase the range to 65535 but precludes the use of negative values. This in turn required the restructuring of the algorithm's calculations in such a way as to prevent negative values from occurring. Where the potential for negative values could not be avoided, a set of if/else statements was used to predict if the resulting value of a calculation step would be negative and the flow of the algorithm was altered to account for this. Additionally, because only integer values are used, division calculations (such as slopes) had to be oriented in such a way to produce a large integer result rather than a small fraction. Lastly, the interpolation and translation of the measurement value to the desired scale sometimes involved very small values where the fractional component is important. A simply change of magnitude unfortunately does not work for this because at other times the value calculated would exceed the unsigned integer bounds if scaled up. To solve this, a dynamically calculated scaling factor is used

which inversely depends on the difference between the two scaled calibration values. This permits small values in subsequent steps to increase in magnitude enough to not lose significant precision while preventing large values from being scaled beyond the unsigned integer upper bound of the microcontroller.

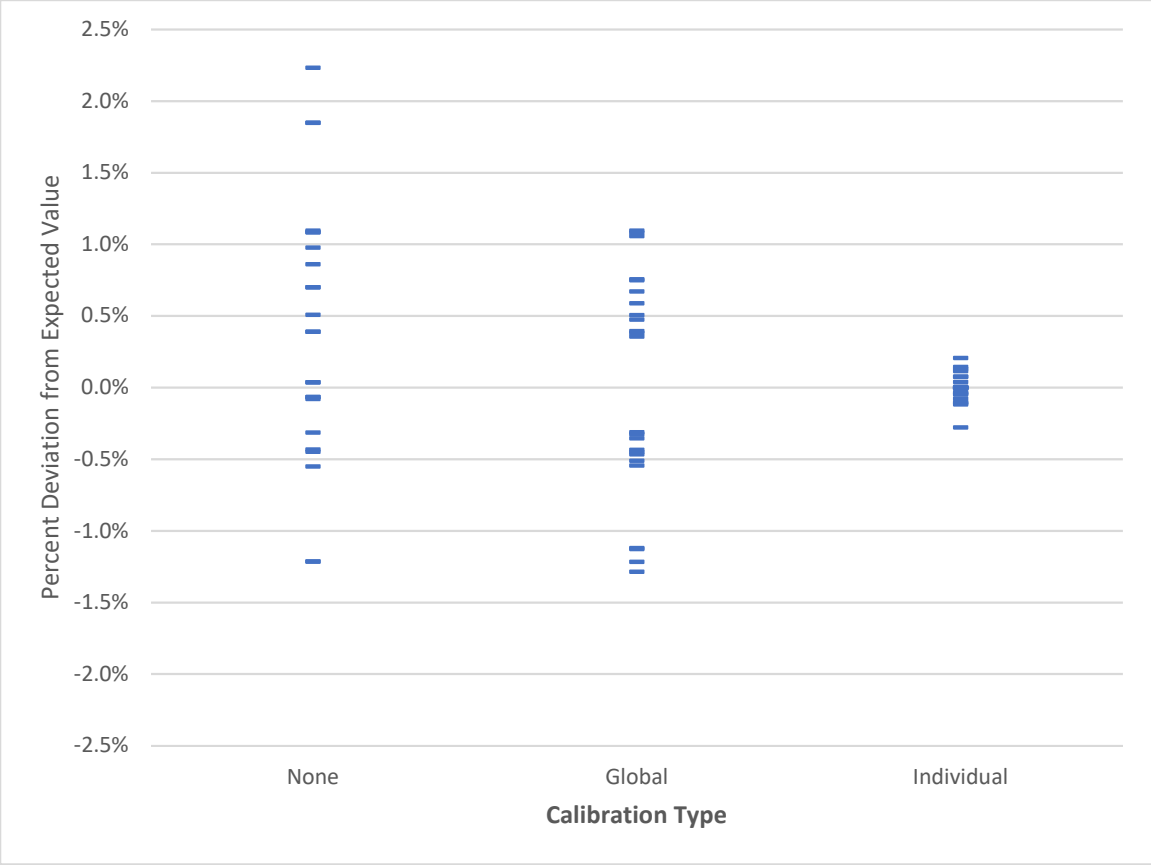

Figure 3-19: Effects of battery compensating calibration

Figure 3-19 shows the effects of different calibrations on the raw measurement values over the sample range of capacitance (47-180pF) and battery voltage (2.6-3.0V). Without any calibration applied, the raw measurements varied with a standard deviation of approximately 1% with an average offset of +0.5%. Applying a global based calibration, that is averaging offsets across different sensor and tag combinations, reduced the average

measurement offset to effectively 0%. However, the standard deviation remained close to the case with no calibration applied. The optimal case is calibrating each sensor and tag combination individually, which yielded not only an average offset of 0% but all measurement values remained within less than 0.5% of their expected value.

4 Experimental Setup, Data, and Analysis

4.1 Parameter Optimization

To maximize battery life, several parameters determining the characteristics of the sensing platform must be optimized. Simply reducing the current draw during sensing does not necessarily translate to a lower power draw. Additionally, the desired precision of measurements places a lower bound on how far energy consumption can be reduced before it has a detrimental effect on sensor resolution. In similar fashion to the energy measurements, fixed value capacitors were used to help determine optimal parameters for the value of the resistor and the time keeping clock frequency. Many measurements were taken over long periods (on the order of hours) to determine the effects of varying these parameters with respect to signal noise, measurement precision, and energy usage. Additional tests were conducted to study the dependence of the MSP430's onboard oscillator and the radio chip's crystal oscillator on varying supply voltages (to simulate battery aging) and environmental temperatures. This was deemed necessary since a stable oscillator is needed for reliable time keeping which affects the accuracy of sensor measurements.

4.1.1 Noise Reduction Through Multiple Samples

One of the first parameters to be optimized was the number of measurement samples to be taken and averaged during a single sensing period. Since the variance of uncorrelated noise while measuring a stable quantity is constant, averaging over n samples reduces the noise power by a factor of $1/n$ [Figure 4-1]. In the case of four samples, the

effect of noise is reduced to 25% of its single sample case. Most sensor measurements rarely varied by more than a few counts, so averaging four samples effectively stabilized the measured values. Additionally, in cases where a raw sensor measurement oscillated between two adjacent integer values, averaging over multiple measurements yielded a rounding toward one or the other depending on which one the true value tended to be closer to.

As described earlier, the MSP430 microcontroller's comparator has a feature to swap its inputs. Doing so allows the inherent offset between the comparator's inputs to be cancelled out. Odd numbers of samples would not cancel out this offset as effectively, therefore only samples in multiples of two (except for the single sample case) were considered.

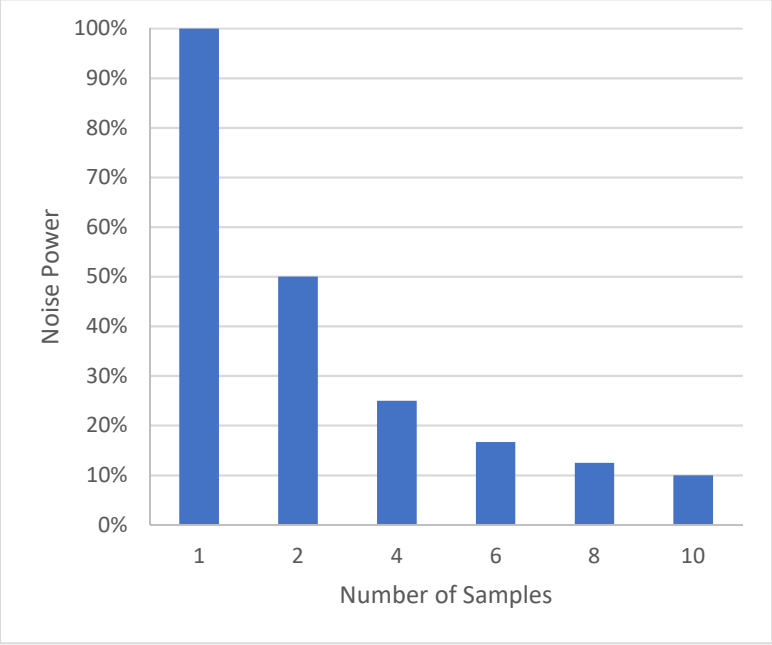

Figure 4-1:Relative noise power as a function of number of measurements

It is evident from Figure 4-1 that averaging over more samples reduces the effect of noise on the measurement. However, the effectiveness of each additional pair of samples

starts to decline rapidly after four samples, with each additional pair contributing less and less to the noise reduction in a non-linear fashion. Energy usage on the other hand, increases linearly with the number of measurements taken, thereby disproportionately increasing the energy cost for every improvement in noise provided by additional samples. For this reason, averaging over four samples was determined to be optimal because it provided an appreciable reduction in noise power without spending a disproportionate amount of energy to do so.

4.1.2 Energy Use

Energy usage measurements were taken using an oscilloscope to measure the time-series voltage drop across a 10 ohm resistor placed in series with a fixed voltage supply providing the power to the TPIP-K tag and sensor. Using this method, the current draw over time of the system could be precisely monitored. By also monitoring the voltage across the sensor, the period during which measurements were taking place could be easily identified and correlated with the current draw [Figure 4-2].

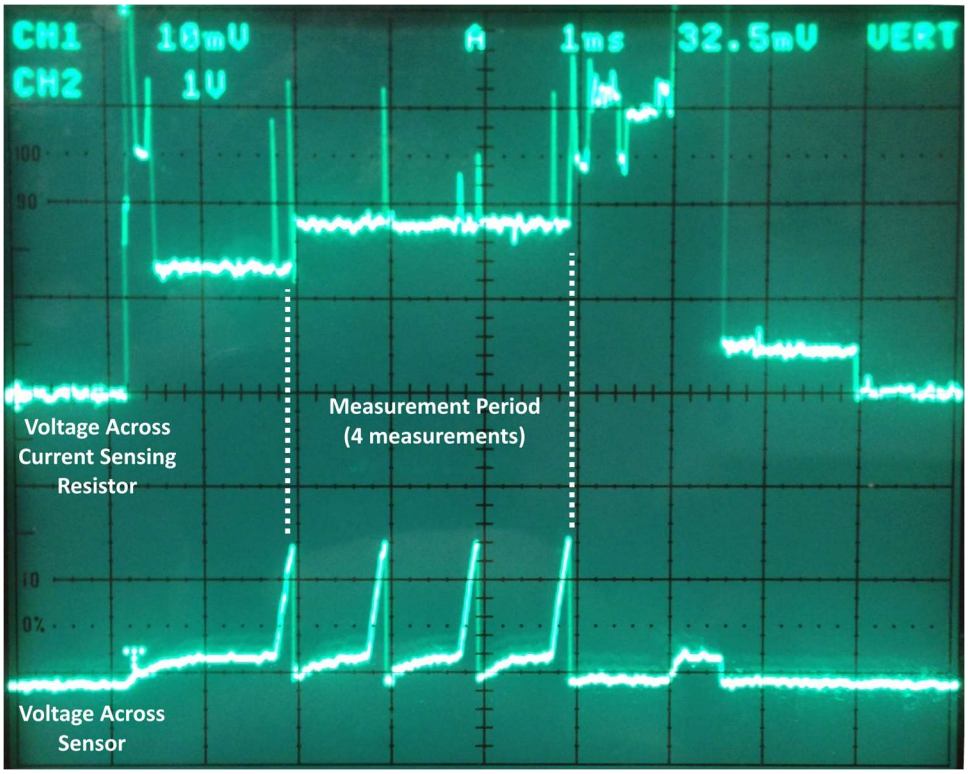

Figure 4-2: Energy measurement

A simple integral of the measurement period multiplied by the known fixed supply voltage yields the approximate amount of energy used during measurement. To reduce the number of unknown variables, temperature stable fixed value capacitors (±5% tolerance) were used in place of the sensor that corresponded to known values in the range of possible capacitances the sensor could have when installed in soil. This range (approximately 47-180 pF) was determined by taking rough measurements of the sensor when dry and when submerged in water using a multimeter in capacitance mode.

Given that there are multiple variables that can be tweaked to not only affect measurement precision but also per measurement energy consumption, extensive data was taken to best characterize these effects and provide enough information to determine optimal parameter selection for a real world use case. The first step in characterizing the

energy use of sensor measurements was to look at the relationship between the current consumption during measurement versus the clock frequency used for time keeping which determines the precision of the measurements. As is shown in Figure 4-3 the relationship is fairly linear for a fixed capacitor and resistor value. While a polynomial fit to the data points slightly better, the added complexity for subsequent calculations was deemed unnecessary for the < 1% improvement in error.

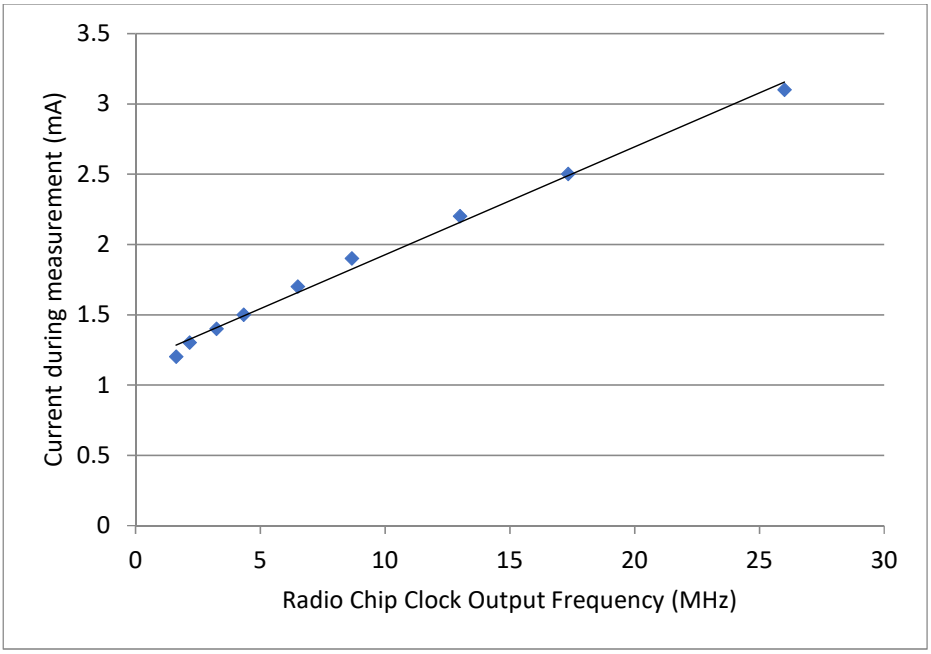

Figure 4-3: Current during measurement at various radio chip clock frequencies

As discussed earlier, the timer module of the MSP430 present on the TPIP-K was discovered to be quite capable of functioning reliably even at clock frequencies far beyond specification. However, to eliminate the possibility of anomalies in real world use the calculations were restricted to frequencies under 16 MHz (the maximum system clock frequency officially specified for the MSP430G2553). This limitation also improves the accuracy of the linear estimate over the range of 2.16MHz - 13MHz to be 1% on average

(0-2% over the range). Outside the prescribed frequency range, the error in estimates increases to 3-6%.

$$current\ during\ measurement\ (mA) = 1.14 + \frac{timer\ clock\ frequency\ (MHz)}{12}$$

Equation 4-1

Using Equation 4-1 and the fact that changes in the value of the sensor circuit resistance and capacitance have a proportional effect on the time for the sensor to reach ½ Vcc [Equation 3-11], a deterministic relationship can be established between the energy consumed to take a single measurement and the three parameters: sensor capacitance, sensing circuit resistance, and timer clock frequency. Using a combination of empirical data and simulation, the energy required to perform a complete soil moisture measurement can be predicted. Additionally, the optimal set of system parameters, sensing circuit resistance and timer clock frequency, can be determined to provide the desired measurement precision while at the same time minimizing energy use.

$$E = nV_{cc}I_{measurement}t_{measurement}t_{wait}$$

Equation 4-2

Using Equation 4-2, the energy usage of a set of measurements can be determined. In this equation, n represents the number of samples taken, V_{cc} is the supply voltage, $I_{measurement}$ is the current consumption during a measurement [Equation 4-1], $t_{measurement}$ is the time it takes to complete a single sample (Equation 3-11 solved for t), and t_{wait} is the brief time (20µs) taken between measurements to allow the sensor probe to discharge fully. Performing this calculation across the available clock frequencies and charging resistor values for the lowest and highest expected sensor capacitance values yields charts like Figure 4-4 and Figure 4-5 which can be used to select values for f and R

that minimize energy usage without sacrificing measurement resolution (discussed in the next section).

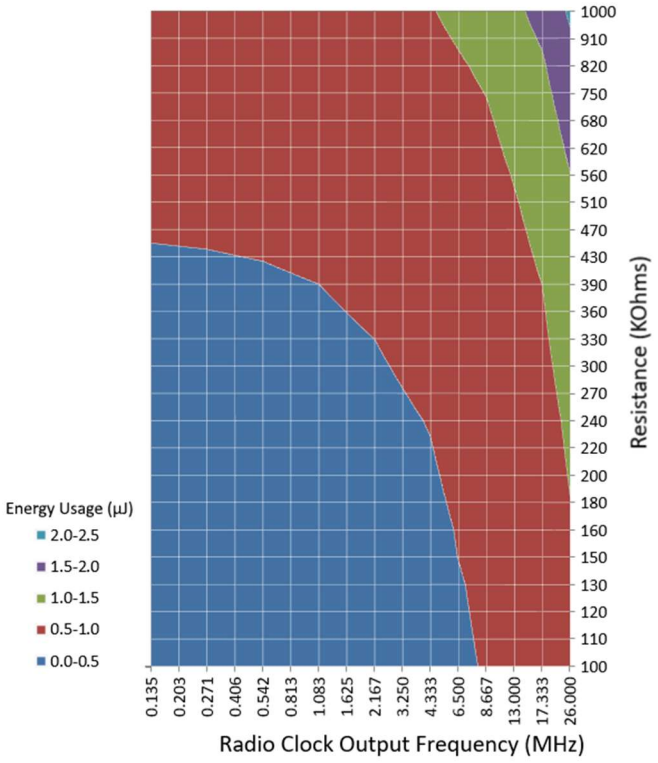

Figure 4-4:Energy usage for 4 measurements at low capacitance, 50 pF (i.e. low moisture content)

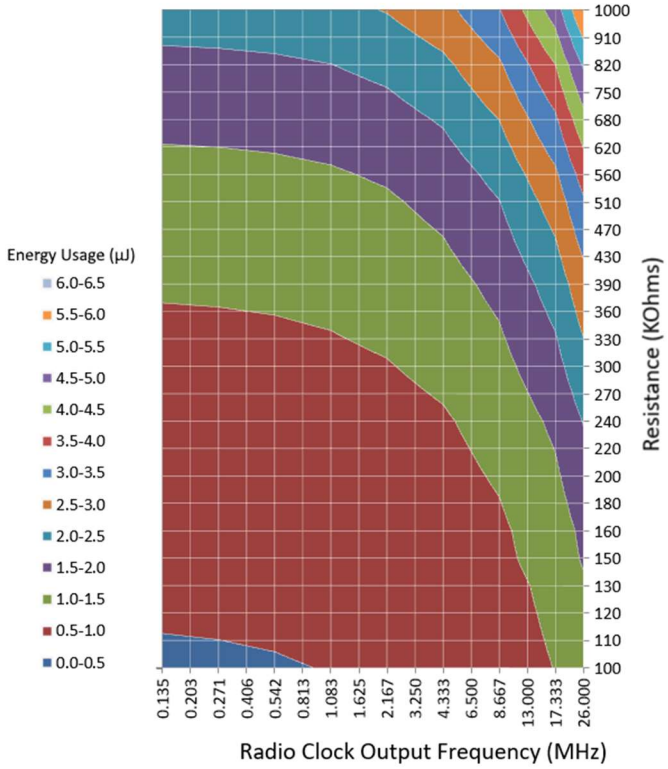

Figure 4-5:Energy usage for 4 measurements at high capacitance, 200pF (i.e. high moisture content)

4.1.3 Measurement Resolution

Just as determining the number of samples to take has a significant impact on the energy required to perform a measurement, so do the parameters that affect how much energy is used per sample. The known resistance *R* determines how quickly the sensor probe "charges up" to the ½ Vcc measurement trigger point. A reduction in *R* translates to a quicker charge rate for the sensor probe, which reduces the time it takes to make a measurement and therefore reduces overall energy consumption. All else being equal, this reduces available resolution as well since the number of countable units of time is reduced leading to cases where two similar measurements end up being "rounded" up to the next nearest countable unit of time. The radio chip's clock frequency *f* is also directly proportional to the resolution with which a measurement can be taken. For any specific

probe capacitance, a higher clock frequency to the Timer_A module will yield a greater count value than if a lower frequency were used, thereby giving finer resolution to the measurement. However, as seen from the prior energy use experiments, higher clock frequencies consume more energy and can also cause various components to operate outside of their design parameters.

To select the optimal values for *R* and *f*, it is important to understand how these parameters relate to the measurement value. Rewriting Equation 3-11 to determine the time it takes for the sensor probe to reach ½ V_{cc} and multiplying that by the number of Timer_A clock ticks per second at a known frequency *f*, we get Equation 4-3 which can be used to predict the counter value of Timer_A for a specific sensor probe capacitance, where *f* is radio's clock output frequency in MHz, *R* is the charging resistor value in kΩ, and *C* is the minimum or maximum expected capacitance in pF..

$$Timer\_A\ \ count = \frac{-f \ln\left(\frac{1}{2}\right) RC}{1000}$$

Equation 4-3

Assuming dry and saturated sensors have an approximate capacitance of 50 pF and 200 pF respectively, the expected count values can be calculated for all possible combinations of selectable radio clock output frequency (135 kHz to 26 MHz) and standard resistor values (100 Ω to 1 MΩ). Using the calculated count values for the dry and saturated sensor readings, the maximum resolution (as percent of measurement) over the range of sensor measurements can be calculated.

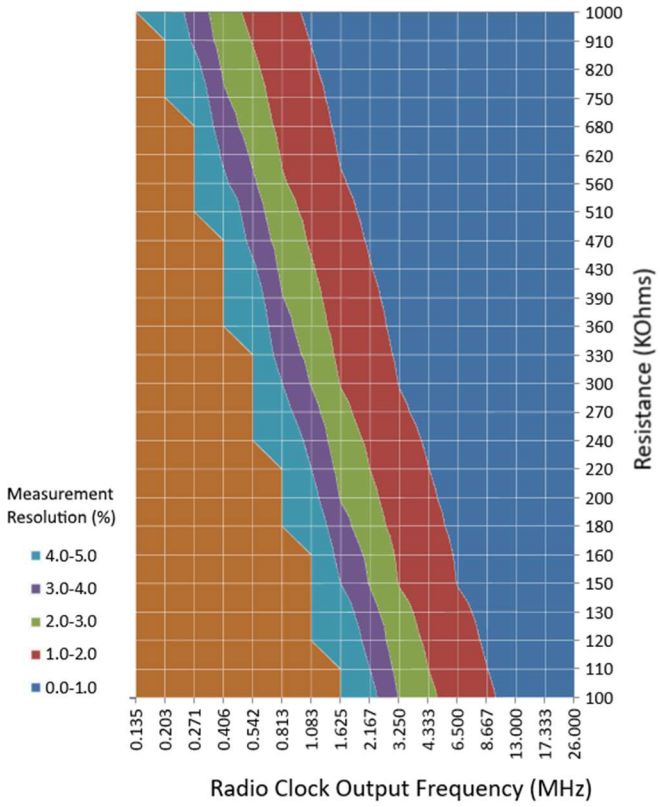

Figure 4-6: Percent resolution over the range of sensor capacitance (50-200 pF)

As expected, increasing either the clock frequency *f* or the resistance *R* results in a linear improvement in measurement resolution. However, the resistance of *R* cannot be increased to an arbitrarily convenient value for two significant reasons. The first reason is that the amount of energy used for a single measurement increases proportionally with increase in time it takes to perform a measurement (e.g. a doubling of the *R* value, doubles the time it takes to take a measurement, which doubles the energy used). Careful analysis and planning must be used to justify the increase in energy use for more measurement resolution which does nothing to lower the noise floor of the measurements. Additionally, an apparent limit on the maximum value of *R* was observed. Empirical testing using fixed value capacitors in place of the moisture probe showed this to be around 1MΩ. For $R <$ 1MΩ, multiple consecutive measurements followed a normal distribution. However, for *R*

> 1MΩ, the measurements began to follow a bimodal distribution whose separation between the peaks increased as the value of *R* was increased. While the readings could still be averaged to a value close to that when $R < 1M\Omega$, the increase in statistical noise necessitated an order of magnitude more measurements to be averaged, which is not optimal in terms of energy usage. Likewise, as was seen in the prior section on energy usage, increasing the clock frequency *f* results in an increase in energy usage although not as severely as changing the *R* value. For these reasons, careful selection of these parameters is necessary to maximize the battery life of the sensor.

4.2 Effects of Salinity

As stated before, water will naturally dissolve natural minerals and salts present in the soil. The salts, which lead to corrosion of exposed electrodes in some sensor designs, also have a discernible effect on the measurements taken with the capacitive style sensor. Testing was conducted by placing the bare sensor PCB in a series of salt water baths with varying concentrations of salt and recording the measured values. The upper bound for salinity was set to approximately 35 ppt in accordance with the average salinity of sea water [22]. There is a clear and dramatic effect with even a small (0.5 ppt) amount of salt dissolved in the water, and the effect increases with higher concentrations [Figure 4-7]. However, the recommended maximum salinity for agricultural irrigation is between 0.5 and 3 ppt depending on crop [23]. In this range, the difference in the effect of different salinities on the measurement is on the order of about 1% and can thus be considered negligible.

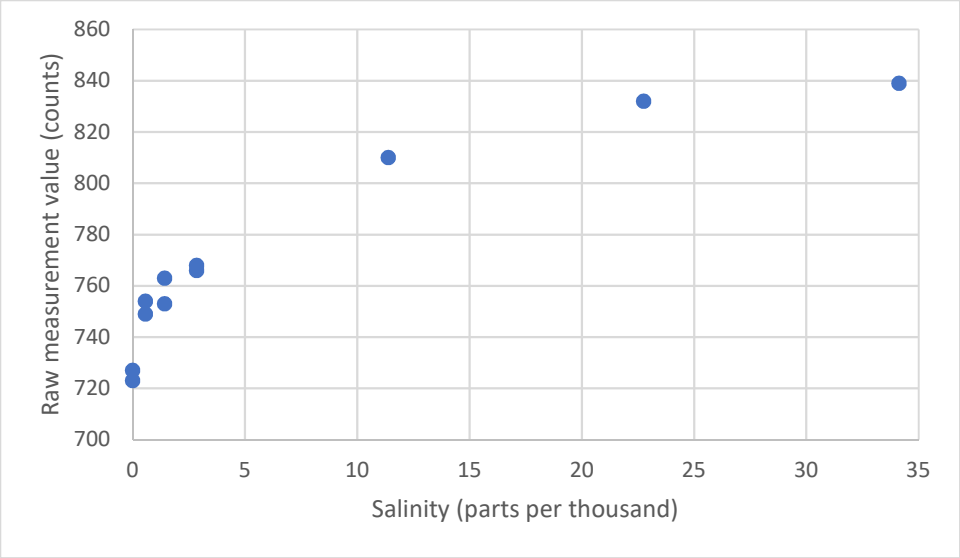

Figure 4-7: Salinity vs. measurement value

4.3 In-Soil Experiments

4.3.1 Test Rig

The in-soil testing rig was designed with an attempt to replicate real world conditions while allowing for relatively fast responding, repeatable, and controlled experiments. The testing rig, seen in Figure 4-8 and Figure 4-9, featured a small sealable chamber, containing a measured amount of sample soil along with bare PCB and packaged moisture sensors. This inner chamber was set inside a larger outer chamber used to collect excess water runoff. The bottom of the smaller chamber was made of the same 8 mil filter paper used on the senor probe to allow excess water not absorbed by the soil or sensor to freely exit into the outer chamber. This was done to mimic a real installation where any water not soaked up by the soil near the moisture sensor would be soaked up by the soil around and below, limiting the volumetric water content to the particular soil's field capacity. Both chambers had air tight lids sealing their interiors from the surrounding

environment to prevent evaporated water inside from getting out, and moist air outside from entering the test setup. This was done to prevent the surrounding environment from affecting measurement of the soil's capacity for water, which is used for determining relative percentages of soil moisture through calibration of the sensor. Additionally, the small air volume of the chambers would result in relatively negligible evaporation, allowing for accurate determination of the soil's water content.

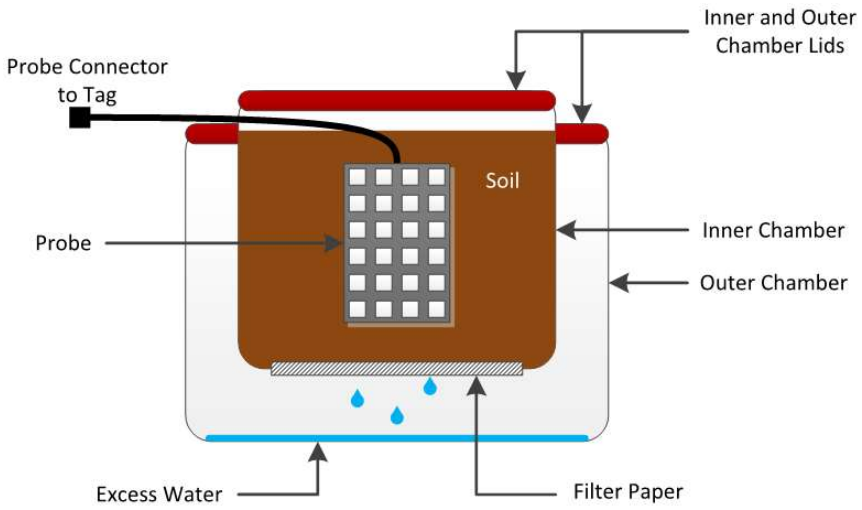

Figure 4-8: In-soil sensor probe testing rig design

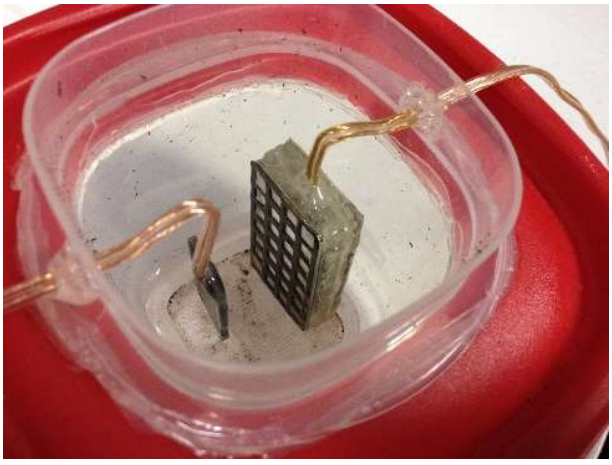

Figure 4-9: In-soil testing rig with bare and completed sensor probes

4.3.2 Soil Types

Two different soil types, acquired from a local home improvement store, were used during testing to assess the sensor's performance in a simulation of the different soil environments it may be installed in. The first was an all-purpose topsoil (Earthgro Topsoil) which was a fine powder when dried out and contained no visible wood chips, rocks, or additives. The second was a potting soil (Vigoro Organic Potting Mix) which had a relatively high content of wood chips, small rocks, and fertilizer pellets. The differences between the soils affect both their density and porosity which in turn should result in different water content values.

4.3.3 Drying Procedure

The soil samples used were baked in a drying oven at 60°C for 6 hours before each experimental run in the progressive wetting test rig. This was deemed sufficient as the soil came out a very fine powder that did not clump (unlike soil which was exposed to room air humidity for extended periods of time) and longer baking times did not result in lower initial measurement readings during sensor testing.

To allow for easy modification and repair during experimentation, the completed sensor package was held together using a thermoplastic adhesive (hot glue). Due to the relatively low melting point of the adhesive, the sensor could not be dried using the drying oven method as was done for the soil samples without compromising the integrity of the sensor package. To facilitate short drying times without resulting in damage, the sensor was suspended in a sealed container at room temperature above a layer of anhydrous crystalized calcium chloride ($CaCl_2$). The calcium chloride readily absorbed the moisture

from the air inside the container which resulted in water from the sensor evaporating to take its place. The small volume of the container and large amount of calcium chloride used prevented the formation of a saturated salt solution which would have limited the humidity inside the drying container to 32%RH. Instead, because the calcium chloride remained in crystal form, it absorbed much more moisture from the air resulting in less than 16%RH which was the bottom measurement limit of the hygrometer used for monitoring the process. As such, the sensor was dried very quickly (< 1 hour) to measurement readings similar to that of a unpackaged sensor board in free air while resulting in no damage to the sensor package. Interestingly, when the sensor was later exposed to room temperature air at 50%RH, the measurement readings would increase slightly over the course of several hours as the diatomaceous earth absorbed some moisture from the surrounding air. This confirms the desirable hygroscopic properties of the diatomaceous earth.

4.3.4 Wetting Procedure

Testing was performed by first baking out the soil samples in a drying oven at 60°C for a minimum of 6 hours to remove water. This was deemed sufficient because when the soil was placed in a sealed test rig with a dry moisture probe installed and allowed to acclimate for 24 hours, no appreciable change in the moisture measurement was detected. After oven drying, the soil sample was then equally distributed by volume (108 cm^3) between two test rigs, each with a moisture probe centered horizontally with approximately 1cm of soil above and 0.5cm of soil below the probe. The dry weight of each complete rig setup was recorded and the rigs were allowed to acclimate in 20°C room temperature for

24 hours. After acclimation, moisture measurements were taken using three different TPIP-K tags at four different battery voltages to confirm calibration functions were working correctly and to have as many data points as possible with the limited available hardware. Following this, 15mL of distilled water were added to each test rig and moisture measurements were recorded for 24 hours using one tag per probe. This process was repeated until addition of more water to the test rigs resulted in excess water dripping down into the lower chamber of the test rigs indicating the soil samples were at full saturation (i.e. field capacity). The amount of excess water was measured and subtracted from the known amount of water added to calculate the volumetric water content at each stage of the experiment. This was then correlated to the raw moisture measurement data from the probes.

4.3.5 Results

After following the drying procedures, the sensors were put into the test rig with a dry-baked soil sample. Since both the sensor and soil readily absorb moisture from the surrounding air, the setup of the sealed test rig was performed quickly immediately following the respective drying procedures to minimize this effect. Repeated tests showed no appreciable change in the lower bound measurements despite small inconsistencies in setup time leading to the conclusion that no significant amount of moisture was absorbed from the air. Since the soil sample under test was at a significantly elevated temperature upon removal from the drying oven, the assembled and sealed test setup was left to equilibrate to room temperature (approximately 20°C) over several hours.

As previously stated, the procedure for the controlled progressive wetting of the soil samples involved the addition of 15mL of water every 24 hours. In this way, a known amount of water was added at each step resulting in very little excess water (approximately 2 ml for either soil sample) dripping into the lower chamber of the test setup once the soils reached their saturation points. With minimal excess water, the whole process could be divided into nearly equal and predictable steps, roughly every 25% of field capacity for the potting soil and every 20% for the top soil.

For a measured soil sample volume of 108 cm^3, the potting soil reached saturation after the addition of 57-58ml of water, resulting in a field capacity of 52% volumetric water content. An equal in volume sample of top soil reached saturation after the addition of 70-73ml of water, resulting in field capacity of 65% volumetric water content. This significant increase in field capacity is likely due to the absence of small rocks and wood chips in the top soil that were present in the potting soil.

Averaging the dry soil measurements across multiple sensors, TPIP tags, soil samples, and battery voltages yielded a capacitance of 50 pF ($\sigma < 2$ pF). Performing the same averaging for the fully saturated soil measurements resulted in a capacitance value of 200 pF ($\sigma < 8$ pF). These average values were the source for the 50 pF and 200 pF capacitance values used in other testing procedures, such as those on energy use.

In both the dry and saturated scenarios, most measurements were within <4% of the average. However, when looking at the complete set of measurements from the intermediate levels of moisture, it is obvious that without individual calibration, the measurements are practically useless because any raw measurement can be mapped to a

wide range of volumetric water content values and could be off by as much as 40-50 percentage points from the real VWC of the soil (Figure 4-10).

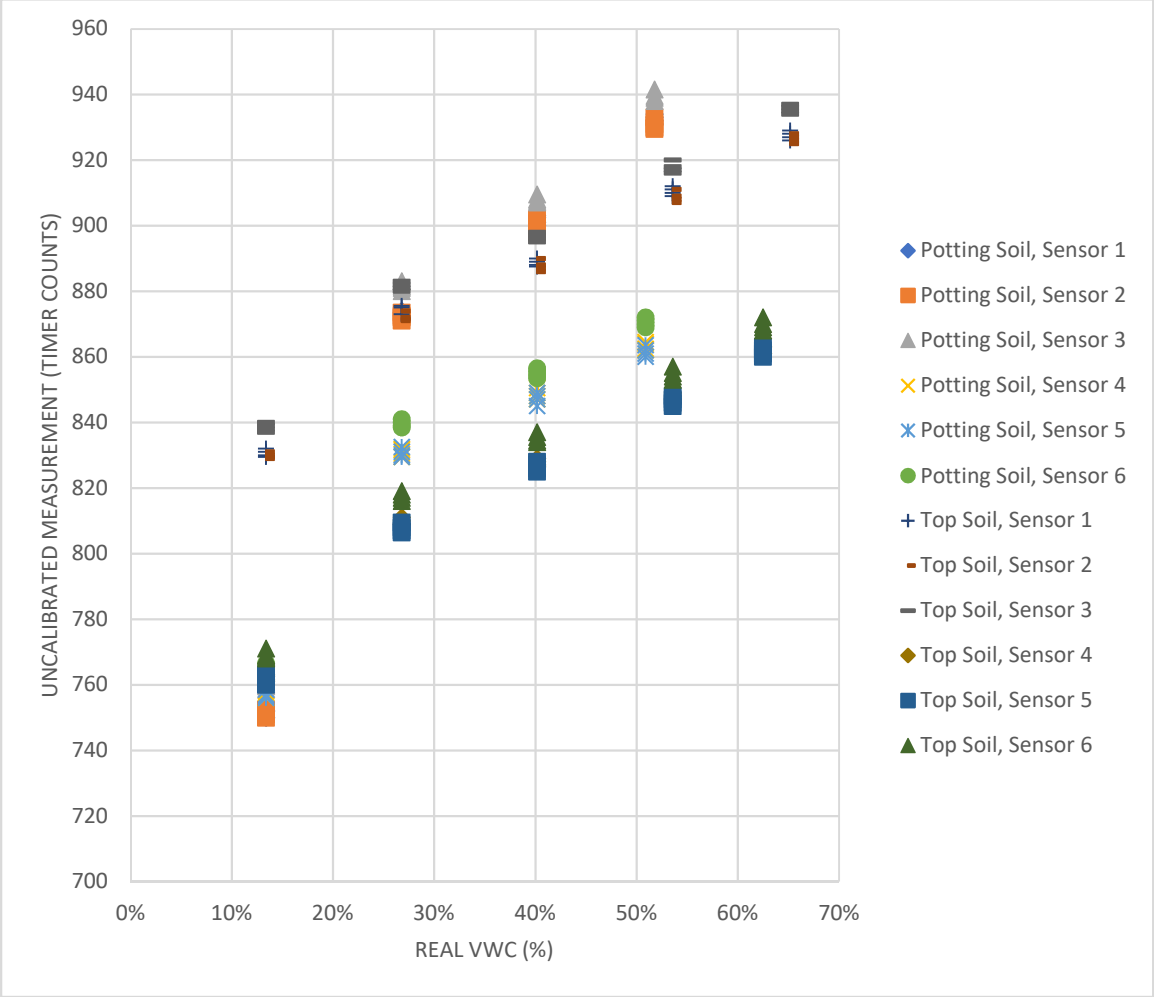

Figure 4-10: Volumetric water content vs raw measurement

Applying the bilinear interpolation based calibration discussed earlier, the apparent lack of correlation between measurements and real VWC can be solved. The most important part in this procedure is the proper selection of calibration points. The field capacity of the soil is naturally selected for the "upper" calibration points because beyond this point, water is no longer being held by the soil. Instead the soil is becoming oversaturated, resulting in soil runoff and generally too wet an environment for plants. This

is also the point at which irrigation should not be applied because the water will just be wasted. The “lower” calibration points on the other hand should ideally be placed at what is known as the permanent wilting point, which is the volumetric water content below which permanent damage will occur to plants. Interestingly enough, the permanent wilting point is approximately half the field capacity regardless of soil type [24], making it’s determination fairly simple when the field capacity for the particular soil is known.

Using this information, the upper calibration points were selected to be the VWC corresponding to field capacity for each soil (52% for the potting soil and 65% for the top soil). The lower calibration points were selected to be the VWC corresponding to approximately half of the field capacity (27% for the potting soil at 53% field capacity and top soil at 41% field capacity). While the lower points are the same due to the lack of granularity during the measured addition of water, the calibration still holds well.

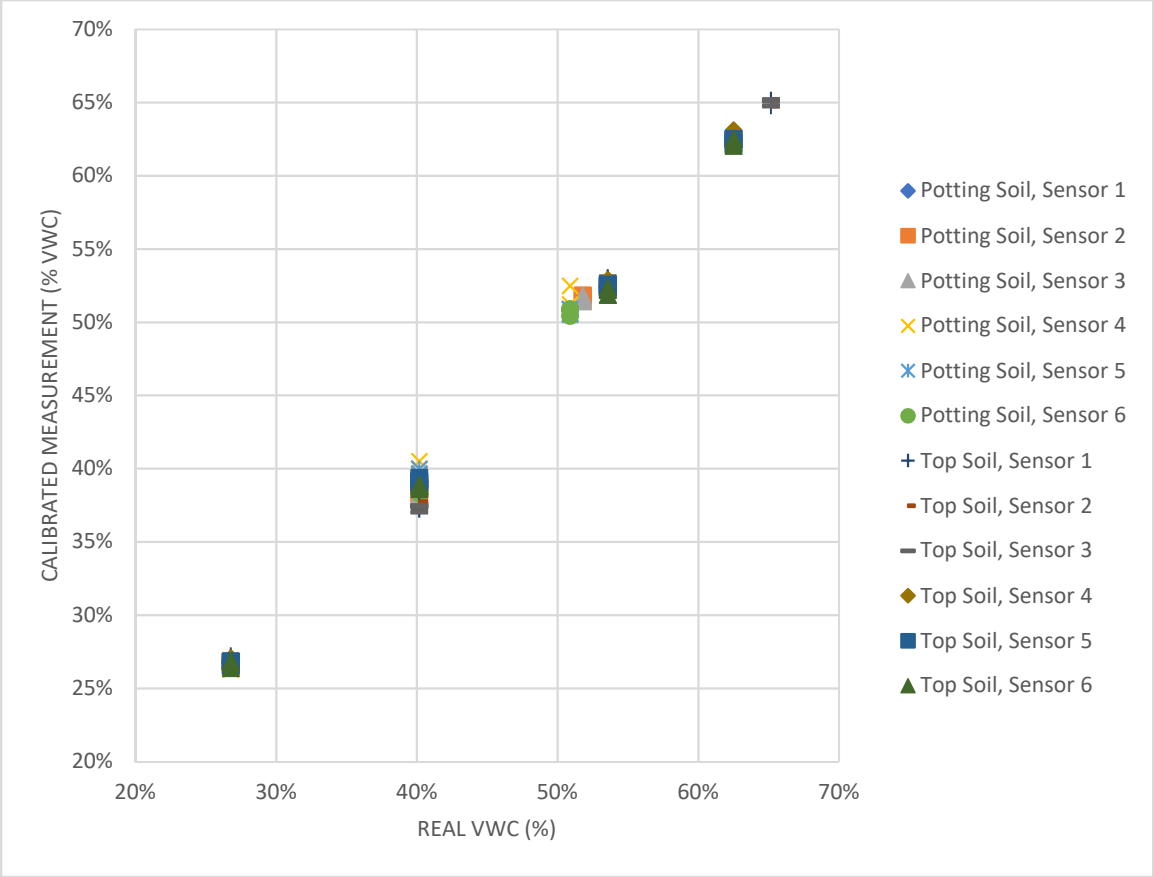

Figure 4-11: Volumetric water content vs VWC calibrated measurement

Applying per sensor individualized calibrations yielded a very linear (better than 3%) relationship between the now calibrated measurement values and the real value of soil VWC [Figure 4-11]. Additionally, effects on measurement resulting from differences between the soil types are compensated for as well, yielding a sensor that reports volumetric water content measurements irrespective of the soil type it is installed in. These results were better than expected and could possibly be further optimized if the lower calibration points were at the VWC corresponding to the permanent wilting point. Nevertheless, the bilinear interpolation pays off measurably and is what makes this sensing solution workable.

5 Conclusion

5.1 Summary

The proposed design and subsequent proof of concept implementation for a low power wireless soil moisture monitoring probe does indeed work as can be seen from the experiment data, providing a measurement accuracy of better than 3% for the volumetric water content of soil within the range of greatest interest between permanent wilting point and field capacity. By using commodity PCB production to manufacture the key element of the sensor probe, a high degree of precision and repeatability was achieved without the costs typically associated with custom tooling and precision manufacturing. Additionally, the TPIP-K platform proved to not only be beneficial from a low-energy standpoint, but the microcontroller features permitted the measurement algorithm to be implemented in hardware instead of software, allowing for extremely precise and repeatable measurements to be taken.

When comparing cost, the design presented in this thesis is more expensive than the low cost, though not reliably accurate, battery-style and resistance based devices, but still within the same order of magnitude. On the other hand, the accuracy of measurements, once calibrated, is comparable to the orders of magnitude more expensive solutions, which require calibration as well. Perhaps the biggest benefit is the potential to leave these sensors installed for extended periods of time and without the need for maintenance, a key differentiator between this and alternative solutions.

The research and development work involved in this project are not limited to soil moisture monitoring. Applications of the fundamental components of this design, both

hardware and software, are readily transferable to other use cases requiring precise measurement of small capacitances in a low power wireless manner. Possible areas include presence detection, discrete value measurement, and fluid level monitoring.

5.2 Future Work

There are several areas where improvements can be made to the soil moisture probe proposed in this paper. More consistent manufacturing methods or alternative designs would result in a smaller difference in measurement values between different sensor probes in the same environment. For example, the current manual application of conformal coating results in inconsistent surface thickness on the sensor probe PCBs which drastically affects the sensitivity and performance of the sensors.

The sensor package can and should be revised to reduce the time constant, which was on the order of hours, by significantly reducing the volume of diatomaceous earth around the sensor PCB. In addition, the filter material used as a barrier between the diatomaceous earth and surrounding soil can be optimized to reduce impedance to liquid water and water vapor movement into and out of the sensor, thereby decreasing the time constant further. Alternative media to diatomaceous earth could also be explored to potentially reduce the time constant or even increase the range of measurement values since the sensor is, by design, measuring the water content of the diatomaceous earth as a function of the water content of the surrounding soil.

Additionally, the software can be revised to enhance measurement accuracy and range. Application of non-linear calibration curves could improve the sensor's accuracy outside the current range of permanent-wilting-point to field capacity. Mathematical models could

also be applied to predict the final measurement value using time constants and the apparently exponential response curves of the sensor.

6 Appendices

These appendices contain the source code that was added to the original TPIP-K firmware to attain the desired functionality for this project. Some sections are in abridged format for brevity because the relevant code was part of a much larger source file, the entirety of which is unnecessary to display here.

6.1 main.c (abridged)

Note: Code is abridged to show only relevant sections written specifically for this project.

```
...
119     void main(void) {
...
132             // Send moisture level
133             header.moisture = ENABLE_MOISTURE;// 0 turns off moisture sensing. 1 turns on moisture sensing.
...
168             // Number of times to wake up before checking moisture
169             int moisture_counter = 1;
...
233             for (;;) {
234
235                     // Decrement the various counters
...
242                     --moisture_counter;
...
269                     if (!(header.decode)) {
...
304                             if (header.moisture) {
305                                     // Do 16-bit moisture sensing, leaving the
306                                     //   data in the dynamic memory pool
307                                     //Do not do moisture sensing every epoch.
308                                     //Use moisture_counter (count down) to determine how often.
309                                     //MOISTURE_STEPS is moisture interval/wake interval in settings.h
310                                     //doMoistureSense in moisture.c
311                                     //True does sensing and resets counter
312                                     //False returns cached value
313                                     if (moisture_counter) {
314                                             doMoistureSense(false);
315                                     } else {
316                                             moisture_counter = MOISTURE_STEPS;
317                                             doMoistureSense(true);
318                                     }
319                             }
320                     }
...
```

6.2 moisture.h

```
1       /*
2        * moisture.h
3        *
4        *  Created on: Apr 23, 2014
5        *      Author: romoore
6        */
7
8       #ifndef MOISTURE_H_
9       #define MOISTURE_H_
10
11      #include "../CC110x/definitions.h"
```

```
12      #include "../CC110x/tuning.h"
13      #include "msp430.h"
14      #include "../settings.h"
15      #include "battery.h"
16
17      extern uint16_t cached_battery;
18
19      /*
20       * Radio Crystal as Timer0_A clock source (TA0CLK)
21       * Author: Jakub Kolodziejski
22       *
23       * Allows the use of the CC1150 radio's crystal oscillator as the clock source
24       * TA0CLK on P1.0 (GDO0) for Timer0_A. Crystal oscillator frequency is 26MHz but
25       * output to TA0CLK can be divided to desired frequency.
26       */
27      void crystalOn(uint8_t divider);   // Wakes up radio and enables output of desired frequency to P1.0 (TA0CLK)
28      void crystalOff();                 // Stops crystal oscillator output from radio and puts radio to sleep
29
30      /*
31       * Moisture Sensing Probe
32       * Author: Jakub Kolodziejski
33       */
34      // Pins used for moisture sensing probe
35      #define CHARGE_PIN  BIT1      // P1.1, Tag AUX pin 8
36      #define SENSE_PIN   BIT2      // P1.2, Tag AUX pin 9
37      // Function definitions
38      void doMoistureSense(bool);
39      void cm_initMoistureSense(void);
40      uint16_t cm_getMoistureLevel(void);
41
42      #endif /* MOISTURE_H_ */
```

6.3 moisture.c

```
1       /*
2        *          moisture.c
3        *
4        *  Created on: Apr 23, 2014
5        *      Author: Jakub Kolodziejski
6        *
7        *
8        *  Capacitive Moisture Sensing
9        *
10       *          Use a capacitive moisture sensor, either a commercial unit or any that uses the high dielectric
11       *          constant of water to determine moisture level. Works also with air humidity sensors that act on
12       *          the same principle.
13       *
14       *
15       *          --- TPIPK1.1 Circuit Design ---
16       *
17       *                R       probe
18       *            +---/\/---+----||----+
19       *            |         |         |
20       *          charge     sense      GND
21       *           pin        pin
22       *
23       *          Description                 MSP430 Pin                  Tag AUX Pin
24       *          Ground                      GND                             1
25       *          CHARGE_PIN                  1.1                             8
26       *          SENSE_PIN                   1.2                             9
27       *
28       *          A resistor "R" (SMD size 0603 or 0804) is placed between pins 8 and 9 of the tag's AUX connector.
29       *          The moisture sensor is connected between pin 9 and pin 1 (GND) of the tag's AUX connector.
30       *          If the moisture sensor is polarized, connect its ground to pin 1 and its positive side to pin 9.
31       *
32       *          For the circuit board soil moisture sensor (range 45-200pF), the R can be 200-400kOhms depending
33       *          on Timer_A clock frequency, energy usage, noise, and resolution needed.
34       *
35       *          DO NOT use an R value greater than 1MOhm because the resulting readings will no longer fall on a
```

```
36          *       normal distribution. Instead the values will start to form two separate peaks (two offset humps).
37          *
38          *
39          *       --- How It Works ---
40          *
41          *       Short Version:
42          *       Measure the time it takes to charge the probe from ground to a reference voltage (i.e. 1/2 Vcc).
43          *       Take and average multiple measurements (to reduce effecots of noise) in multiples of 2 (to remove
44          *       Comparator related offsets). Translate the average measurement value based on calibration
45          *       data and desired scale (i.e. 0-100% moisture content) before transmitting.
46          *
47          *       Long Version:
48          *       1)      The configuration state of all pins and modules to be used is saved ("original" state)
49          *       2)      Pins and modules to be used are configured appropriately.
50          *       3)      CHARGE_PIN and SENSE_PIN are shorted to ground to discharge the sensor quickly.
51          *       4)      SENSE_PIN is reconfigured for sensing using the onboard comparator.
52          *               CHARGE_PIN remains grounded to keep sensor discharged.
53          *       5)      To minimize the effects of nominal battery voltage and temperature, Timer_A's clock is sourced
54          *               from the CC1150 radio's crystal via GDO0. The frequency is selected using
CM_CRYSTAL_DIVIDER.
55          *               See "settings.h" for configuration and list of available frequencies.
56          *       6)      Timer_A is set to start counting up from near its maximum allowed value
(CM_MEASUREMENT_TIME)
57          *               and sets the CHARGE_PIN high when it rolls over back to 0. This synchronizes the "start" of
58          *               measurement to the moment charging of the moisture sensor beings.
59          *       7)      Timer_A is started, CHARGE_ PIN is set high, sensor is being charged, MSP430 is put into LPM_3
60          *               (LPM_1 if using the DCO instead of the radio's crystal)
61          *       8)      When the Comparator detects the voltage across the probe has reached 1/2 Vcc (based on internal
62          *               reference), it triggers a capture of Timer_A's current count to TA0CCR1 and wakes up the MSP430.
63          *               The internal reference can be set to 1/4 Vcc if using capacitance values larger than a few
64          *               hundred pF to ensure the measured region is fairly linear and to reduce measurement time which
65          *               is also limited by the maximum value Timer_A can count to (65535 clock counts).
66          *       9)      The value of TA0CCR1 is added to a running total and the counter of the number of measurements
67          *               taken so far (measurement_count) is incremented. If the number of measurements already taken
68          *               is less than CM_NUM_OF_MEASUREMENTS then go to step #3 (Comparator inputs are swapped
on every
69          *               measurement to remove the effects of Comparator offset when measurements are averaged in
70          *               multiples of 2).
71          *       10) Turn off the radio and its crystal clock output through GDO0
72          *       11) The desired number of measurements has been taken (as defined by CM_NUM_OF_MEASUREMENTS in
73          *               "settings.h"). Divide the running total by the number of measurements taken to get the average.
74          *       12) The average measurement value is translated using calibration data for measurement count
75          *               vs. the nominal battery voltage in mV (as measured and saved into cached_battery). This is then
76          *               further translated to a scale defined for the calibration data (i.e. lower 2 points correspond
77          *               to 0% and upper 2 points to 100% moisture level). The scale values can be anything, such as 25%
78          *               and 75% for a humidity sensor, or even actual capacitance values in pF.
79          *       13) All pins and MSP430 modules used are returned to their "original" state as saved in step #1.
80          *       14) The finalized and scaled measurement is then saved to the transmit buffer.
81          *
82          *
83          *       --- Calibration & Scaling ---
84          *
85          *       Soil moisture:
86          *       Use sensor in soils of known moisture content.
87          *
88          *       Humidity:
89          *       Use saturated salts in enclosed container.
90          *
91          *       How It Works:
92          *       A bilinear interpolation function is used to remove offsets in the measurement values
93          *       and to translate the resulting measurement onto a desired scale (eg. 0 to 100% humidity). There
94          *       exist offsets between different pairs of tags and sensors due to differences in tag/sensor
95          *       manufacture as well as those inherent in the different functional modules of the MSP430 such as
96          *       the comparator and the measurement of Vcc.
97          *
98          *       The calibration depends on 4 data points, each corresponding to a known number of timer counts at
99          *       a known nominal battery voltage (realistically this is the voltage across the large C3 capacitor on
100         *       the TPIP-K 1.1). These 4 data points are split into 2 pairs. The first pair corresponds to a high
101         *       capacitance value of the sensor close or equal to the top of the desired measurement scale. The
102         *       second pair corresponds to a low capacitance value of the sensor close or equal to the bottom of the
```

```
103     *       desired measurement scale. For each pair, one point will correspond to the number of timer counts at
104     *       a relatively "low" battery voltage (eg. 2.2V) while the other will correspond to the number of timer
105     *       counts at a relatively "high" battery voltage (eg. 3.2V) for the same real capacitance value of the
106     *       sensor (i.e. whatever the sensor's capacitance is at 25% and 75% RH). Additionally, each pair is also
107     *       defined with a particular scale value that it corresponds to (eg. 25% RH for one pair and 75% RH for
108     *       the other).
109     *
110     *       1)      Calculate the negative inverse of the slope for each calibration point pair. Negative inverse is
111     *               necessary to maintain a positive large value so as not to lose precision in later calculations,
112     *               this also allows the use of unsigned variables in calculation steps.
113     *       2)      Calculate the "y-axis intercept" if a line were made through a point pair and the y-axis
114     *               corresponded to the number of counts.
115     *       3)      Having a known battery voltage (stored in cached_battery by doBatterySense()) calculate the
116     *               points on the calibration pair lines (AB,CD in the figure below) that would correspond to that
117     *               battery voltage.
118     *       4)      Calculate the calc_scaler factor which prevents loss of precision in calculation steps that would
119     *               otherwise result in very small numbers where the fractional component is important. Also, it
120     *               prevents overflowing the bounds of unsigned integers in steps where very large numbers result.
121     *       5)      Calculate the relative position of the averaged measurement count with respect to the two points
122     *               found in step #3.
123     *       6)      Using this relative position and the known scale position of the calibration points, calculate
124     *               the corresponding position of the measurement value on the measurement scale.
125     *       7)      Return the value resulting after calibration and scaling is applied.
126     *
127     *           |
128     *           |       high capacitance
129     *         c |     A------------B
130     *         o |
131     *         u |
132     *         n |
133     *         t |    low capacitance
134     *         s |     C------------D
135     *           |
136     *           |_____________________________
137     *         0     |           |
138     *               2200        3200
139     *                 voltage in mV
140     *
141     *
142     *       Corresponding to some known position on the desired scale (i.e. 75%) = CM_BIG_C_SCALED
143     *
144     *       A = at a known "low" battery voltage, how many counts does the high capacitance correspond to
145     *         = (CM_BIG_C_LO_BATT, CM_BIG_C_LO_COUNT)
146     *
147     *       B = at a known "high" battery voltage, how many counts does the high capacitance correspond to
148     *         = (CM_BIG_C_HI_BATT, CM_BIG_C_HI_COUNT)
149     *
150     *       Corresponding to some known position on the desired scale (i.e. 25%) = CM_SMALL_C_SCALED
151     *
152     *       C = at a known "low" battery voltage, how many counts does the low capacitance correspond to
153     *         = (CM_SMALL_C_LO_BATT, CM_SMALL_C_LO_COUNT)
154     *
155     *       D = at a known "high" battery voltage, how many counts does the low capacitance correspond to
156     *         = (CM_SMALL_C_HI_BATT, CM_SMALL_C_HI_COUNT)
157     *
...
164     *
165     *       See "settings.h" for parameter definitions and configuration.
166     */
167
168     #include "moisture.h"
169
170     uint16_t cached_moisture = 0;     // the saved value of the previous measurement
171     uint8_t* moisture_location;                     // pointer to the location used in the transmit buffer for moisture data
172
173     /* For moisture sensing code to take control of Timer0_A1 interrupt vector
174      *                      0 allows normal code execution in Timer0_A1 interrupt vector
175      *                      1 runs moisture probe code in Timer0_A1 interrupt vector
176      *
177      * DO NOT MODIFY!!! SHOULD ALWAYS BE SET TO ZERO HERE!!!
```

```
178         */
179         bool sensingMoisture = 0;
180
181         /*
182          *          Moisture Sensing Probe (main function)
183          *          Author: Jakub Kolodziejski
184          *
185          *          Sense moisture level (capacitance) from probe and save value in transmit buffer.
186          *
187          *
188          *          Call to doMoistureSense(true) performs a measurement and saves a new measurement value in the
189          *          transmit buffer.
190          *
191          *          Call to doMoistureSense(false) puts the cached value of the last performed measurement in the
192          *          transmit buffer.
193          */
194
195         void doMoistureSense(bool update) {
196                     // If called with update=true, perform a measurement
197                     if (update) {
198                                 cached_moisture = 0; // Considered an "invalid" value
199
200                                 // *** Save state of all registers modified by the moisture sensing code ***
201                                 uint16_t original_TA0CCR0 = TA0CCR0;
202                                 uint16_t original_TA0CCR1 = TA0CCR1;
203                                 uint16_t original_TA0CCTL0 = TA0CCTL0;
204                                 uint16_t original_TA0CCTL1 = TA0CCTL1;
205                                 uint16_t original_TA0CTL = TA0CTL;
206                                 uint16_t original_CACTL1 = CACTL1;
207                                 uint16_t original_CACTL2 = CACTL2;
208                                 uint16_t original_CAPD = CAPD;
209                                 uint16_t original_P1DIR = P1DIR;
210                                 uint16_t original_P1IE = P1IE;
211                                 uint16_t original_P1OUT = P1OUT;
212                                 uint16_t original_P1REN = P1REN;
213                                 uint16_t original_P1SEL = P1SEL;
214                                 uint16_t original_P1SEL2 = P1SEL2;
215                                 uint16_t original_P2DIR = P2DIR;
216                                 uint16_t original_P2IE = P2IE;
217                                 uint16_t original_P2OUT = P2OUT;
218                                 uint16_t original_P2REN = P2REN;
219                                 uint16_t original_P2SEL = P2SEL;
220                                 uint16_t original_P2SEL2 = P2SEL2;
221
222                                 // Initialize probe related pins and hardware
223                                 // Note that the TimerA clock which is used for the measurement is sourced from the XTAL on the
CC1150.
224                                 // It can be sourced from SMCLK if desired--but there are issues of temperature and Vcc variation
unless the MSP has a crystal of its own.
225                                 cm_initMoistureSense();
226
227                                 // Measure moisture level
228                                 // Returns average counts with compensation for Vcc and optionally for capacitance offset.
229                                 cached_moisture = cm_getMoistureLevel();
230
231                                 // *** Restore all registers modified by the moisture sensing code back to their previously saved
state ***
232                                 TA0CTL = original_TA0CTL;
233                                 TA0CCTL0 = original_TA0CCTL0;
234                                 TA0CCTL1 = original_TA0CCTL1;
235                                 TA0CCR0 = original_TA0CCR0;
236                                 TA0CCR1 = original_TA0CCR1;
237                                 CACTL1 = original_CACTL1;
238                                 CACTL2 = original_CACTL2;
239                                 CAPD = original_CAPD;
240
241                                 P1IE = original_P1IE;
242                                 P1DIR = original_P1DIR;
243                                 P1REN = original_P1REN;
244                                 P1SEL = original_P1SEL;
```

```
245                             P1SEL2 = original_P1SEL2;
246                             P1OUT = original_P1OUT;
247
248                             P2IE = original_P2IE;
249                             P2DIR = original_P2DIR;
250                             P2REN = original_P2REN;
251                             P2SEL = original_P2SEL;
252                             P2SEL2 = original_P2SEL2;
253                             P2OUT = original_P2OUT;
254                     }
255                     // Save value to transmit buffer
256                     moisture_location = malloc(2);
257                     moisture_location[0] = ((uint8_t*) &cached_moisture)[1];
258                     moisture_location[1] = ((uint8_t*) &cached_moisture)[0];
259
260             }
261
262             /*
263              * Moisture Sensing Probe (comparator & charge to 1/2 Vcc based)
264              * Author: Jakub Kolodziejski
265              *
266              *       R       probe
267              *   +---/\/\---+----||----+
268              *              |         |         |
269              *             charge     sense     GND
270              *              pin       pin
271              *
272              *             Description                     MSP430 Pin                      Tag AUX Pin
273              *             Ground                          GND                                     1
274              *             CHARGE_PIN                      1.1                                     8
275              *             SENSE_PIN                       1.2                                     9
276              *
277              *             For mositure sensor, R ~ 200-400kOhms depending on clock frequency, energy usage, and resolution needed.
278              *
279              *             CHARGE_PIN and SENSE_PIN defined in "sensing.h"
280              *
281              *             NOTE: Comment out the call to opticalReceive(&params); in SendBeacon.c, otherwise there is an extra current
draw
282              *                 of ~0.5mA during subsequent individual measurements (seems to be a conflict in usage of Timer0_A).
283              *                 Check status of this before further development.
284              */
285
286             // Initialize probe related pins and hardware
287             void cm_initMoistureSense(void) {
288
289                     // *** Initialize probe related pins ***
290                     // Configure Charge pin...
291                     P1IE &= ~CHARGE_PIN;        // Disable interrupt
292                     P1REN &= ~CHARGE_PIN;       // Disable pullup/pulldown resistor
293                     P1DIR |= CHARGE_PIN;        // Set as output
294                     P1SEL |= CHARGE_PIN;        // Set to TA0.0 output (TA0.0. is used controls charge/discharge via
Timer0_A)
295                     P1SEL2 &= ~CHARGE_PIN;      // ^ see above ^
296                     P1OUT &= ~CHARGE_PIN;       // Not necessary for functionality, just keeps the pin low in case it is not
configured as TA0.0
297                     CAPD &= ~CHARGE_PIN;        // Enable GPIO buffer on CHARGE pin
298                                                                             // When configure pin to be comparator,
disconnect GPIO buffer automatically
299                                                                             // When reconfigure as I/O pin, need buffer
configured.
300                                                                             // This line guarantees that it has the buffer
properly configured.
301                     // Configure Sense pin...
302                     P1IE &= ~SENSE_PIN;                     // Disable interrupt
303                     P1REN &= ~SENSE_PIN;        // Disable pullup/pulldown resistor
304                     P1DIR |= SENSE_PIN;                     // Set as output
305                     P1SEL &= ~SENSE_PIN;        // Set as GPIO
306                     P1SEL2 &= ~SENSE_PIN;       // ^ see above ^
307                     P1OUT &= ~SENSE_PIN;        // Set low, for quick discharge of capacitor between measurements
308                     CAPD &= ~SENSE_PIN;                     // Enable GPIO buffer on SENSE pin.
```

```
309                                                                     //Because it is set output low with no
pulldown, it "instantly" (t < 50ns) discharges the capacitor
310
311                     // *** Configure Comparator_A+ for capacitor measurement ***
312                     CACTL1 = CAREF_2;                       // 1/2 Vcc ref on + pin
313                     CACTL2 = P2CA2 + CAF;       // Input CA2 on - pin, filter output
314
315                     // *** Configure Timer0_A for capacitor measurement ***
316                     TA0CTL = TACLR;  // Stops Timer0_A and resets all parameters including TAR, clock divider, and count
direction
317
318                     //Below is setup for using the XTAL on the radio through GDO0
319                     #if CM_USE_CRYSTAL
320                                     TA0CTL = TASSEL_0 + ID_0 + MC_0;// Use TA0CLK, no division, stopped mode
321             #else
322                     TA0CTL = TASSEL_2 + ID_0 + MC_0;     // Use SMCLK, no division, stopped mode
323             #endif      // CM_USE_CRYSTAL?
324                     TA0CCTL0 = CCIS_2;// CCIS_2 set to GND, set OUTMOD_0, OUT bit to 0 (discharges capacitor through
CHARGE pin)
325                     TA0CCTL1 = CCIS_1 + SCS + CAP;// Use CCI1B (comparator output) for capture/compare trigger,
synchronize capture, set to capture mode
326                     TA0CCTL2 = CCIS_2;// CCIS_2 set to GND, all else cleared to prevent unexpected things from happening
327                     TA0CCR0 = CM_MEASUREMENT_TIME;// Maximum number of counts to permit for a measurement before
it is considered a failure (too high capacitance or resistance) NOTE: Actual time is affected by the clock frequency used for Timer0_A
328                     TA0CCR1 = 0;// Clear capture register to prevent unexpected effects on first measurement
329                     TA0CCR2 = 0;                            // Clear to prevent unexpected things from happening
330             }
331
332             // Measure moisture level
333             uint16_t cm_getMoistureLevel(void) {
334
335             #if CM_USE_CRYSTAL
336                     crystalOn(CM_CRYSTAL_DIVIDER);// Turn ON radio with crystal oscillator outputting desired frequency on
GDO0 to P1.0
337             #endif      // CM_USE_CRYSTAL?
338                     // *** Take control of Timer0_A1 ***
339                     sensingMoisture = 1;  // Set to 1 to take control of Timer0_A1
340
341                     // Prepare measurement taking variables
342                     uint16_t measurement_count = 0;                 // Number of measurements taken
343                     uint16_t measurement_sum = 0;// Sum of measurements taken (to be averaged)
344
345                     // Perform multiple measurements and record them into measurement_sum
346                     while (measurement_count < CM_NUM_OF_MEASUREMENTS) {
347
348                             // *** Prepare for charging and subsequent measurement following time
CM_MEASUREMENT_WAIT ***
349                             //      Reconnect the SENSE pin to the Comparator (following the use of the SENSE pin to
quickly discharge the capacitor)
350                             //      Configure Comparator for measurement
351                             //      Configure Timer0_A for charge control and count measurement
352                             CAPD |= SENSE_PIN;// Disable GPIO buffer on SENSE pin (this is just a precaution, according to
the User's Guide this is unnecessary if a pin is defined as a Comparator input using P2CAx bits)
353                             CACTL2 |= P2CA2;                                        // Reconnect SENSE pin to
Comparator
354                             CACTL1 ^= CAEX;   // Swap Comparator inputs and invert it's output (averaging 2^n successive
measurements with this swap in between will cancel out the Comparator's offset voltage)
355                             CACTL1 |= CAON;                                                 // Turn on
comparator
356                             TA0R = 0xFFFF - CM_MEASUREMENT_WAIT;// Offset Timer0_A starting point to wait a given
amount of counts before first measurement is taken (Used to ensure capacitor is discharged). NOTE: Actual time is affected by the
clock frequency used for Timer0_A
357                             TACCTL1 |= CM_2 + CCIE;     // Set Timer_A1 capture on falling edge, enable interrupt
358                             TACCTL0 = OUTMOD_1;         // Set CHARGE pin high on next "overflow" to start charging the
capacitor at the same time as the Timer starts counting from 0
359
360                             // *** Start Timer0_A ***
361                             //      While the timer is running, the capacitor is first held discharged (CHARGE pin is low)
for CM_MEASUREMENT_WAIT counts.
```

```
362                             //          When the timer overflows, the CHARGE pin is set high and the Comparator will trigger a
Timer count capture to TA0CCR1 when the voltage across the capacitor reaches Vcc/2
363                             //          This method synchronizes the Timer counting from 0 to the start of capacitor charging,
eliminating the possiblility of software delay offsets.
364                             TA0CTL |= MC_1;                         // Start Timer0_A in UP mode (count to TA0CCR0)
365
366                             // *** Sleep CPU while "discharge wait" and "charge/measurement" occur ***
367         #if CM_USE_CRYSTAL
368                             _BIS_SR(LPM3_bits + GIE);
369                             // Enter low power mode (LPM3) w/ CPU interrupts
370         #else
371                             _BIS_SR(LPM1_bits + GIE);
372                             // Using DCO, must stay in LPM1 or the SMCLK is shut down.
373                             // Enter low power mode (LPM1) w/ CPU interrupts
374         #endif      // CM_USE_CRYSTAL?
375                             // Waiting for capacitance to be measured...
376                             //          (when the Comparator triggers at Vcc/2 on the SENSE pin, Timer0_A1 interrupt captures
the measurement count and wakes up the CPU for normal code execution)
377                             //  Note that the interrupt used is the same as used for the light sensor code.
378                             // *** Discharge capacitor ***
379                             //          The CHARGE pin is pulled low to discharge and hold the capacitor at ground until the
next measurement period begins.
380                             //          The SENSE pin is disconnected from the Comparator and pulled low to "instantly" (t <
50ns) discharge the capacitor.
381                             TA0CCTL0 = OUTMOD_0;// Sets CHARGE pin low immediately to discharge capacitor
382                             CACTL2 &= ~P2CA2;// Disconnect SENSE pin from Comparator for quick discharge of capacitor
when GPIO buffer is enabled
383                             CAPD &= ~SENSE_PIN;         // Enable GPIO buffer on SENSE pin. Because it is set output low
with no pulldown, it "instantly" (t < 50ns) discharges the capacitor
384
385                             // *** Record measurement ***
386                             measurement_count++;                    // Increment measurement counter
387                             measurement_sum += TA0CCR1;             // Add current measurement to sum total
388                             TA0CCR1 = 0;// Clear capture register to prevent unexpected effects on next measurement
389                 }
390
391                 // *** All measurements completed ***
392                 //          Timer0_A and Comparator are already turned off after the last measurement is taken
393                 //          Capacitor is already discharged after the last measurement is taken
394                 //          Reset all remaining hardware and pins used for measurement
395         #if CM_USE_CRYSTAL
396                 crystalOff();                                   // Turn OFF radio crystal oscillator on P1.0 (GDO0)
397         #endif      // CM_USE_CRYSTAL?
398                 sensingMoisture = 0;  // Release control of Timer0_A1
399
400                 TA0CTL = TACLR;                                 // Resets all parameters of Timer0_A including TAR, clock
divider, and count direction
401
402                 // *** Process measurements ***
403                 uint16_t measurement = 0;                       // Measurement value. Initialize to zero.
404
405                 // Average the measurements to remove comparator offset and reduce noise contributions
406                 measurement = measurement_sum / CM_NUM_OF_MEASUREMENTS;     // Take average of measurements
407
408                 // Bilinear Interpolation Calibrated Correction for Measurement Counts vs Battery Voltage
409                 /*
410                  * This section makes use of 2-point calibration data for a low and a high capacitance (for a total
411                  * of 4 calibration points) to account for the dependence of the measured count value on Vcc.
412                  * Additionally, the result is scaled over a desired range (i.e. 0 to 100 for percentage in 1%
413                  * increments) before being passed on for transmission.
414                  *
415                  * See "settings.h" for parameter definitions
416                  */
417                 /* Calculate the negative inverse of the slope and the intercept of the line defined by
418                  * the upper calibration bound points.
419                  */
420                 uint16_t big_c_slope = (CM_BIG_C_HI_BATT - CM_BIG_C_LO_BATT) / (CM_BIG_C_LO_COUNT -
CM_BIG_C_HI_COUNT);
421                 uint16_t big_c_intercept = CM_BIG_C_LO_COUNT + (CM_BIG_C_LO_BATT / big_c_slope);
422                 /* Calculate the negative inverse of the slope and the intercept of the line defined by
```

```
423                     * the lower calibration bound points.
424                     */
425                     uint16_t small_c_slope = (CM_SMALL_C_HI_BATT - CM_SMALL_C_LO_BATT) /
(CM_SMALL_C_LO_COUNT - CM_SMALL_C_HI_COUNT);
426                     uint16_t small_c_intercept = CM_SMALL_C_LO_COUNT + (CM_SMALL_C_LO_BATT / small_c_slope);
427                     /* Calculate the upper and lower calibration bound points as defined by the above slopes
428                      * and intercepts which correspond to the current known battery voltage (mV).
429                      */
430                     uint16_t big_c_count = big_c_intercept - (cached_battery / big_c_slope);
431                     uint16_t small_c_count = small_c_intercept - (cached_battery / small_c_slope);
432
433                     // Measurement value translated to the scale defined in "settings.h". Initialize to zero.
434                     uint16_t scaled_value = 0;
435                     uint16_t Ptop = 0;
436                     int Pbottom = 0;
437                     uint16_t calc_scaler = 0;
438                     int P = 0;
439
440                     if(measurement == small_c_count) {          // measurement = lower calibration bound
441                             /* Just return the lower bound scaled equivalent because further math is unnecessary
442                              * and this also avoids having to deal with a divide by zero in those calculations.
443                              */
444                             scaled_value = CM_SMALL_C_SCALED;
445                     }
446                     else {      // measurement != lower calibration bound
447                             /* Calculate the normalized position of the measurement within the range defined by
448                              * the upper and lower calibration bounds. The top and bottom half of the equation
449                              * are calculated separately because Ptop is useful for calculating calc_scaler and
450                              * this splitting reduces having to calculate it separately.
451                              */
452                             Ptop = (big_c_count - small_c_count);
453
454                             /* calc_scaler is used in subsequent calculations to maximize precision lost with
455                              * the use of integer variables and math when the results of individual calculation
456                              * steps are very small (<10) and heavily depend on the fractional component of the
457                              * calculations. It is necessary to limit the calc_scaler so that in multiplication
458                              * it does not result in a value greater than +/- 32767 else it will be beyond the
459                              * bounds of an int causing garbage results.
460                              */
461                             calc_scaler = 65535 / Ptop;
462                             /* Calculate the normalized position of the measurement within the range defined by
463                              * the upper and lower calibration bounds.
464                              */
465                             if(measurement > small_c_count) {
466                                     Pbottom = (measurement - small_c_count);
467                                     P = (calc_scaler * Ptop) / Pbottom;
468                                     // Translate the measured value to the desired scale defined in "settings.h"
469                                     scaled_value = CM_SMALL_C_SCALED + (calc_scaler * (CM_BIG_C_SCALED -
CM_SMALL_C_SCALED)) / P;
470                             }
471                             else {      // measurement < small_c_count, NOTE: the equal case is taken care of before
472                                     Pbottom = (small_c_count - measurement);
473                                     P = (calc_scaler * Ptop) / Pbottom;
474                                     // Translate the measured value to the desired scale defined in "settings.h"
475                                     scaled_value = CM_SMALL_C_SCALED - (calc_scaler * (CM_BIG_C_SCALED -
CM_SMALL_C_SCALED)) / P;
476                             }
477                     }
478
479             #if CM_RAW_MEASUREMENT              // Return raw measurement value in timer counts WITHOUT calibration and
scaling applied (useful for testing and manual calibration)
480                     return measurement;
481             #else
482                     // Perform min/max scale limit safety checks and force the value to within the limits if it is out of the lower and
upper bounds (limits set in "settings.h")
483                     if(scaled_value < CM_MIN_SCALED) {
484                             return CM_MIN_SCALED;
485                     }
486                     else if(scaled_value > CM_MAX_SCALED) {
487                             return CM_MAX_SCALED;
```

```
488                 }
489
490                 // Return the measured value with calibration and scaling applied
491                 return scaled_value;
492         #endif
493         }
494
495         /*
496          *          Radio Crystal as Timer0_A clock source (TA0CLK)
497          *          Author: Jakub Kolodziejski
498          *
499          *          Allows the use of the CC1150 radio's crystal oscillator as the clock source TA0CLK on P1.0 for
500          *          Timer0_A. The default crystal oscillator frequency is 26MHz but output to TA0CLK via GDO0 can be
501          *          divided down to a desired frequency (see "settings.h" for proper settings). To wake up the radio
502          *          and enable output of the desired clock frequency on GDO0, call crystalOn(x) where "x" is one of
503          *          the CRYSTAL_DIV_# defined in "settings.h". When finished with using the radio's crystal, call
504          *          crystalOff() to turn off the output to GDO0 and shutdown the radio.
505          *
506          *          Available Frequencies (and their respective dividers):
507          *          NOTE: something like 0.3_ = 0.33333... (repeating decimal)
508          *                    Divider set to 1      ->          f = 26 MHz
509          *                    Divider set to 1.5    ->          f = 17.3_ MHz
510          *                    Divider set to 2      ->          f = 13 MHz
511          *                    Divider set to 3      ->          f = 8.6_ MHz
512          *                    Divider set to 4      ->          f = 6.5 MHz
513          *                    Divider set to 6      ->          f = 4.3_ MHz
514          *                    Divider set to 8      ->          f = 3.25 MHz
515          *                    Divider set to 12     ->          f = 2.16_ MHz
516          *                    Divider set to 16     ->          f = 1.625 MHz
517          *                    Divider set to 24     ->          f = 1.083_ MHz
518          *                    Divider set to 32     ->          f = 812.5 kHz
519          *                    Divider set to 48     ->          f = 541.6_ kHz
520          *                    Divider set to 64     ->          f = 406.25 kHz
521          *                    Divider set to 96     ->          f = 270.83_ kHz
522          *                    Divider set to 128    ->          f = 203.125 kHz
523          *                    Divider set to 192    ->          f = 135.416_ kHz
524          *
525          *
526          *          --- Current Consumption ---
527          *
528          *          When the MSP430 is put into LPM_3 while the radio crystal is on, the sleep current can be estimated
529          *          roughly as follows:
530          *                          GDO0 Frequency (MHz)
531          *     Sleep Current (mA) = 1.16 +  ----------------------
532          *                                  13
533          *
534          *          So for a 13MHz clock on GDO0 (CRYSTAL_DIV_2) the sleep current will be about 2.16mA
535          *          This is only a rough estimate based on collected data fitted with a linear plot. Actual value may
536          *          be off by as much as 0.1mA (0.04mA on average).
537          */
538
539         // Wakes up radio and turns on output of desired frequency to P1.0 (TA0CLK)
540         void crystalOn(uint8_t divider) {
541                 // Configure P1.0 (GDO0) to use radio's crystal oscillator for TA0CLK
542                 P1IE &= ~BIT0;                  // Disable interrupt
543                 P1DIR &= ~BIT0;                 // Set as input
544                 P1REN &= ~BIT0;                 // Disable pullup/pulldown resistor
545                 P1SEL |= BIT0;                  // Set to TA0CLK (TimerA_0 clock source)
546                 P1SEL2 &= ~BIT0;    // ^ see above ^
547                 P1OUT &= ~BIT0;                 // Set low
548                 CAPD &= ~BIT0;                  // Enable GPIO buffer
549
550                 // Configure P2.6 so it doesn't interfere with clock signal to P1.0
551                 P2IE &= ~BIT6;                  // Disable interrupt
552                 P2DIR &= ~BIT6;                 // Set as input
553                 P2REN &= ~BIT6;                 // Disable pullup/pulldown resistor
554                 P2SEL &= ~BIT6;                 // Set to GPIO
555                 P2SEL2 &= ~BIT6;    // ^ see above ^
556                 P2OUT &= ~BIT6;                 // Set low
557
```

```
558                 // Wake CC1101 and wait for oscillator to stabilize
559                 Wake_up_CC1101();
560                 // Turn on oscillator with desired frequency being output to P1.0 (GDO0)
561                 // (typ. current consumption 1.1mA)
562                 TI_CC_SPIWriteReg(TI_CCxxx0_IOCFG0, divider);
563         }
564
565         // Stops crystal oscillator output from radio and puts radio to sleep
566         void crystalOff(void) {
567                 // Reset GDO0 to high impedence tri-state as set by ReWriteCC1101Registers() in main.c
568                 TI_CC_SPIWriteReg(TI_CCxxx0_IOCFG0, 0x2E); // GDO0 output pin to high impedance 3-state
569                 // Turn off crystal oscillator (typ. current consumption 0.22 mA)
570                 TI_CC_SPIStrobe(TI_CCxxx0_SXOFF);
571                 // Power down CC1101 when Csn goes high (typ. current consumption 200nA)
572                 TI_CC_SPIStrobe(TI_CCxxx0_SPWD);
573         }
```

6.4 interrupt.c (abridged)

Note: Code is abridged to show only relevant sections written specifically for this project.

```
...
9           extern bool sensingMoisture;
10          extern uint8_t ambient_val;
11          /*
12           * Interrupt Service Request (ISR) for Timer0_A1-3
13           * Used for ambient light sensing (Timer0_A3)
14           * Used for comparator based moisture sensing (Timer0_A1)
15           */
16          #pragma vector=TIMER0_A1_VECTOR
17          __interrupt void TIMER0_A1_ISR(void) {
18                  // Moisture probe code...
19                  if (sensingMoisture) {
20                          TACCTL1 &= ~(CM_2 + CCIFG);          // Turn off TA1 captures, clear interrupt flag
21                          TACTL &= ~MC_2;                                  // Stop Timer_A
22                          CACTL1 &= ~CAON;                                 // Turn off Comparator
23
24          #if CM_USE_CRYSTAL
25                          LPM3_EXIT;                                       // Exit low power mode (LPM3)
26          #else
27                          LPM1_EXIT;                                       // Exit low power mode (LPM1)
28          #endif      // CM_USE_CRYSTAL?
29                  } else {
...
44                  }
45          }           // TIMER0_A1_ISR
```

6.5 settings.h (abridged)

Note: Code is abridged to show only relevant sections written specifically for this project.

```
...
60                  //Set ENABLE_MOISTURE to 1 if doing sensing.
61          #define ENABLE_MOISTURE 1
...
95          // Interval between moisture sensing
96          #define MOISTURE_INTVL TWO_SECOND
...
104         #define MOISTURE_STEPS (MOISTURE_INTVL/WAKE_INTVL)
...
110         /* Moisture sensing */
111
112         //Use capacitive moisture sensor, either commercial unit or any kind of probe that uses the high dielectric constant of
water.
113         //For TPIPK1.1, one end of the capacitor is ground (pin 1) and the other end is connected to sense pin (9).
114         //Sense pin is also connnected through a resistor to pin 8, which is used to charge the capacitor.
```

```
115	//For circuit board sensor, R~ 200-400kOhms.  Not critical, but can be optimized for noise, resolution, and energy.
116	//First, short capacitor to ground through sense pin and pull charge pin low to keep it low.
117	//Sense pin reset to sensing function; charge pin is kept low.
118	//To minimize effects of temperature and Vcc, TimerA clock is sourced from GD00 set to produce the output of the XTAL on the CC1150.
119	//The XTAL is divided by CM_CRYSTAL_DIVIDER.  A value of 4 gives 6.5 MHz for the clock, which works well for 390 Kohm resistor and soil probe.
120	//Start TimerA preset to a high value and have it count up.
121	//When overflows to zero use that to synchronize the start of the charge cycle.
122	//Charge capacitor with Vcc (circuit board sensor =45 pfd dry and 170 pfd wet) through resistor until voltage reaches the reference voltage (Vcc/2)
123	//as measured by the sense pin using the comparator.
124			//Can choose Vcc/2 or Vcc/4 for reference.
125			//Vcc/4 gives lower resolution, but 1/4 the energy of Vcc/2.
126	//Stop timer and add value to running sum to get average of several measurements.
127	//Discharge capacitor as before and then swap the inputs of the comparator and repeat.
128		//This cancels the input offset voltage of the comparator to first order.
129		//Measurements must be multiples of two.
130	//Note that the fundamental measurement is the time for the voltage to fall to a fraction of Vcc,
131	//and is not sensitive to Vcc to first order.
132		//In practical situations, Vcc drops during a single measurement, making compensation necessary
133		//Vcc compensation is done by measuring the nominal battery voltage and using a calibrated correction factor at each value of Vcc.
134		//Note that the correction factor includes the fact that the voltage drops during a measurement,
135		//so changes in the measurement timing changes the fudge factor.
136		//Note that this is best done using an actual battery since the voltage/time profile depends on the charge state of the battery.
137		//The internal resistance of the battery can vary from about 20 ohms fresh to about 200 ohms near end of life.
138		//Calibrations are done down to 2.2V.
139	//Multiple pairs of measurements are averaged and then the compensation factor is used to correct for Vcc.
140	//A second compensation factor is used to correct for the capacitance of the circuit board, the cable, and the pin on the MSP430.
141		//Could be set to correct for all of the dry capacitance if desired.
142	//Initial code returns raw timer counts (2 Bytes)--needs to use individual calibration values to convert to %RH.
143		//Conversion to %RH will reduce packet length to 1 Byte.
144		//Saturated salts will be used to calibrate humidity sensors.
145		//Soil moisture sensor will use known moisture content.
146
147	// C_Meter moisture sensing related settings
148	// Preferred method of sensing.
149	//		IMPORTANT!!! Make sure that (expected maximum counts to Vcc/2) * NUM_OF_MEASUREMENTS < 65535
150	//		Running total when summing the measurements.
151	#define CM_NUM_OF_MEASUREMENTS		4			// Number of measurements to average, must be a factor of 2 to compensate for Comparator offset
152	#define CM_MEASUREMENT_TIME		0xFFFF	// Max number of timer counts to wait for measurement completion before it is considered invalid (NOTE: affected by clock frequency used for Timer0_A)
153	#define CM_MEASUREMENT_WAIT		130			// Number of timer counts to wait between measurements to ensure capacitor is properly discharged (NOTE: affected by clock frequency used for Timer0_A, pick a value that gives around 20us for the frequency you choose)
154
155	// Bilinear Interpolation Calibrated Correction for Measurement Counts vs Battery Voltage
156	#define CM_RAW_MEASUREMENT		1			// If set to 1, data transmitted will be the raw timer count value without calibration and scaling applied (useful for testing and manual calibration)
157	#define CM_MIN_SCALED			0		// Absolute minimum scale value (anything less than this will be reported as this value)
158	#define CM_MAX_SCALED			500		// Absolute maximum scale value (anything greater than this will be reported as this value)
159	#define CM_SMALL_C_SCALED		47		// Lower cal. points scale value equivalent
160	#define CM_SMALL_C_HI_BATT		3081		// Lower cal. point "high battery voltage" measured battery voltage (mV)
161	#define CM_SMALL_C_HI_COUNT		258	// Lower cal. point "high battery voltage" measured count
162	#define CM_SMALL_C_LO_BATT		2701	// Lower cal. point "low battery voltage" measured battery voltage (mV)
163	#define CM_SMALL_C_LO_COUNT		261	// Lower cal. point "low battery voltage" measured count
164	#define CM_BIG_C_SCALED			180			// Upper cal. points scale value equivalent
165	#define CM_BIG_C_HI_BATT		3081	// Upper cal. point "high battery voltage" measured battery voltage (mV)
166	#define CM_BIG_C_HI_COUNT		848			// Upper cal. point "high battery voltage" measured count
```

```
167     #define CM_BIG_C_LO_BATT            2680    // Upper cal. point "low battery voltage" measured battery voltage
(mV)
168     #define CM_BIG_C_LO_COUNT                   851             // Upper cal. point "low battery voltage"
measured count
169
170
171
172     // Radio Crystal as Timer_A clock source (TA0CLK)
173     //          NOTE: something like 0.3_ = 0.33333... (repeating decimal)
174     #define CRYSTAL_DIV_1               0x30    // Divider set to 1              ->          f = 26 MHz
175     #define CRYSTAL_DIV_1p5             0x31    // Divider set to 1.5    ->          f = 17.3_ MHz
176     #define CRYSTAL_DIV_2               0x32    // Divider set to 2              ->          f = 13 MHz
177     #define CRYSTAL_DIV_3               0x33    // Divider set to 3              ->          f = 8.6_ MHz
178     #define CRYSTAL_DIV_4               0x34    // Divider set to 4              ->          f = 6.5 MHz
179     #define CRYSTAL_DIV_6               0x35    // Divider set to 6              ->          f = 4.3_ MHz
180     #define CRYSTAL_DIV_8               0x36    // Divider set to 8              ->          f = 3.25 MHz
181     #define CRYSTAL_DIV_12              0x37    // Divider set to 12     ->          f = 2.16_ MHz
182     #define CRYSTAL_DIV_16              0x38    // Divider set to 16     ->          f = 1.625 MHz
183     #define CRYSTAL_DIV_24              0x39    // Divider set to 24     ->          f = 1.083_ MHz
184     #define CRYSTAL_DIV_32              0x3A    // Divider set to 32     ->          f = 812.5 kHz
185     #define CRYSTAL_DIV_48              0x3B    // Divider set to 48     ->          f = 541.6_ kHz
186     #define CRYSTAL_DIV_64              0x3C    // Divider set to 64     ->          f = 406.25 kHz
187     #define CRYSTAL_DIV_96              0x3D    // Divider set to 96     ->          f = 270.83_ kHz
188     #define CRYSTAL_DIV_128             0x3E    // Divider set to 128    ->          f = 203.125 kHz
189     #define CRYSTAL_DIV_192             0x3F    // Divider set to 192    ->          f = 135.416_ kHz
190     // Use the CC110x crystal as the Timer_A clock source for moisture measurements
191     //          0 = use SMClock, 1 = use crystal
192     #define CM_USE_CRYSTAL 1
193     // Radio crystal frequency divider to use for moisture measurement
194     // With XTAl of 26 MHz and divider of 4, get 6.5 MHz for TimerA clock.
195     // Be careful to note maximum frequency of MSP450 at the minimum Vcc expected.
196     // Experimentally, we see that the MSP430 can run up to 26 MHz--with a high current.  Reliability not guaranteed.
197     #define CM_CRYSTAL_DIVIDER                  CRYSTAL_DIV_4
198
199     #endif /* SETTINGS_H_ */
```

7 Bibliography

[1] S. Graham, C. Parkinson and M. Chahine, "The Water Cycle," 1 October 2010. [Online]. Available: http://earthobservatory.nasa.gov/Features/Water/.

[2] I. Shiklomanov, "World fresh water resources," in *Water in Crisis: A Guide to the World's Fresh Water Resources*, P. H. Gleick, Ed., New York, Oxford University Press, 1993.

[3] B. Osterath, "Wastewater crop irrigation risks health of nearly a billion people," 7 April 2017. [Online]. Available: https://www.dw.com/en/wastewater-crop-irrigation-risks-health-of-nearly-a-billion-people/a-39538101.

[4] L. Mosley, "Drought impacts on the water quality of freshwater systems; review and integration," *Earth-Science Reviews,* p. 140, November 2014.

[5] The World Bank, "Population, total," [Online]. Available: http://data.worldbank.org/indicator/SP.POP.TOTL. [Accessed 20 March 2018].

[6] Irrometer Company, *WATERMARK Soil Moisture Sensor — MODEL 200SS.*

[7] R. van der Lee, "The Vinduino Project-2: Making and Installing Gypsum Soil Moisture Sensors," 15 July 2012. [Online]. Available: http://vanderleevineyard.com/vineyard-blog/the-vinduino-project-2-making-and-installing-gypsum-soil-moisture-sensors.

[8] A. Martinez and A. P. Byrnes, "Modeling Dielectric-constant values of Geologic Materials," *Current Research in Earth Science,* vol. 247, 2001.

[9] "Measuring moisture content using time-domain reflectometry," 4 March 2018. [Online]. Available: https://en.wikipedia.org/wiki/Measuring_moisture_content_using_time-domain_reflectometry. [Accessed 20 March 2018].

[10] J. A. Huisman, S. S. Hubbard, J. D. Redman and A. P. Annan, "Measuring Soil Water Content with Ground Penetrating Radar: A Review," *Vadose Zone Journal,* vol. 2, pp. 476-491, November 2003.

[11] Wikipedia, "Neutron probe," 23 November 2014. [Online]. Available: https://en.wikipedia.org/wiki/Neutron_probe. [Accessed 20 3 2018].

[12] Skye Instruments, *Soil Moisture Measurement Guidance Notes.*

[13] D. A. Boyarskii, V. V. Tikhonov and N. Y. Komarova, "Model of Dielectric Constant of Bound Water in Soil for Applications of Microwave Remote Sensing," *Progress In Electromagnetics Research,* vol. 35, pp. 251-269, 2002.

[14] I. J. Bahl, Lumped Elements for RF and Microwave Circuits, Norwood, MA: Artech House, 2003.

[15] Texas Instruments, *MSP430G2x53, Mixed Signal Microcontroller, SLAS735J,* 2013.

[16] "Threshold voltage," 8 March 2018. [Online]. Available: https://en.wikipedia.org/wiki/Threshold_voltage#Dependence_on_temperature. [Accessed 20 March 2018].

[17] Z. Albus, *PCB-Based Capacitive Touch Sensing With MSP430, SLAA363A,* Texas Instruments, 2007.

[18] Texas Instruments, *MSP430x2xx Family User's Guide, SLAU144J,* 2013.

[19] Abracon, *Ceramic SMD Crystal,* 2015.

[20] TDK, *Multilayer Ceramic Chip Capacitors,* 2017.

[21] Energizer Brands, LLC, *Energizer CR2032 Product Datasheet.*

[22] F. J. Millero, R. Feistel, D. G. Wright and T. J. McDougall, "The composition of Standard Seawater and the definition of the Reference-Composition Salinity Scale," *Deep Sea Research Part I: Oceanographic Research Papers,* vol. 55, no. 1, pp. 50-72, January 2008.

[23] G. Fipps, "Irrigation Water Quality Standards and Salinity Management Strategies".

[24] P. Trail, O. Abaye, T. L. Thompson and W. Thomason, "Conservation agriculture in Senegal: Comparing the Effects of Intercropping and Mulching on Millet Yields," January 2015.