

INSTRUCTIONS FOR BUILDING A REAL-TIME WELL WATER TEMPERATURE, CONDUCTIVITY & WATER LEVEL METER

Version 1

1 September 2020

Prepared by: John Drage

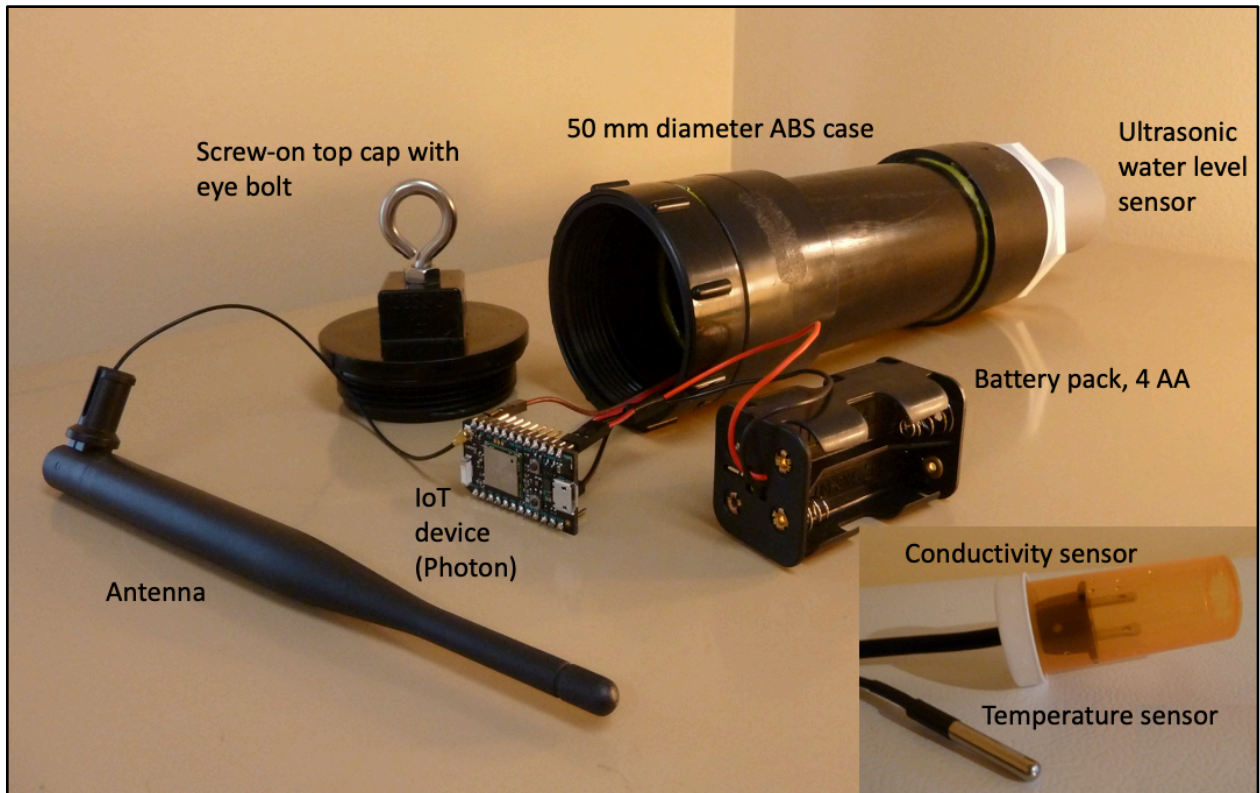


TABLE OF CONTENTS

	PAGE <u>NO.</u>
1.0 Introduction	1
2.0 Parts List.....	4
3.0 Meter Construction.....	6
3.1 Step 1 – Assemble the Meter Case	6
3.2 Step 2 – Attach Wires to Sensors	6
3.2.1 Water Level Sensor	6
3.2.2 Temperature Sensor	8
3.2.3 Conductivity Sensor	10
3.3 Step 3 – Attach Sensors, Battery Pack, and Antenna to IoT Device.....	12
3.4 Step 4 – Software Setup	13
3.5 Step 5 – Test the Meter.....	15
4.0 Meter Construction and Operation Tips	17
5.0 Installing the Meter in a Water Well	18
5.1 Before Going to the Field	18
5.2 Field Setup Procedure.....	19
5.3 Field Installation Tips.....	23
6.0 How to Make a Cellular Version of the Meter	25
8.0 References	27

Appendix A - Conductivity Sensor Verification

Appendix B - Webhook Setup Instructions

1.0 Introduction

This report provides instructions for building a low-cost, real-time, water meter for monitoring temperature, Electrical Conductivity (EC) and water levels in dug wells. The meter is designed to hang inside a dug well, measure the water temperature, EC and water level once a day, and send the data by WiFi or cellular connection to the Internet for immediate viewing and downloading. The cost for the parts to build the meter is approximately Can\$230 for the WiFi version and Can\$330 for the cellular version. The water meter is shown in Figures 1 and 2. A previously published version of this water meter is available for monitoring water levels only (Drage, 2020).

The meter uses three sensors: 1) an ultrasonic sensor to measure the depth to water in the well; 2) a waterproof thermometer to measure water temperature; and 3) a common household two-pronged plug, which is used as a low-cost EC sensor to measure the electrical conductivity of the water. The ultrasonic sensor is attached directly to the meter case, which hangs at the top of the well and measures the distance between the sensor and the water level in the well; the ultrasonic sensor is not in direct contact with the water in the well. The temperature and EC sensors must be immersed under the water; these two sensors are attached to the meter case with a cable that is long enough to allow the sensors to extend below the water level.

The sensors are attached to an Internet-of-Things (IoT) device that connects to a WiFi or cellular network and sends the water data to a web service to be graphed. The web service used in this project is ThingSpeak.com (ThingSpeak, 2019), which is free to use for non-commercial small projects (fewer than 8,200 messages/day). In order for the WiFi version of the meter to work, it must be located close to a WiFi network. Domestic water wells often meet this condition because they are located close to a house with WiFi. The meter does not include a data logger, rather it sends the water data to ThingSpeak where it is stored in the cloud. Therefore, if there is a data transmission problem (e.g. during an Internet outage) the water data for that day are not transmitted and are permanently lost.

The meter design presented here was modified after a meter that was made for measuring water levels in a domestic water tank and reporting the water level via Twitter (Ousley, 2015). The main differences between the original design and the design presented here are the ability to operate the meter on AA batteries instead of a wired power adaptor, the ability to view the data in a time-series graph instead of a Twitter message, the use of an ultrasonic sensor that is specifically designed for measuring water levels, and the addition of temperature and EC sensors.

The low-cost, custom-made EC sensor, which is made with a common household plug, was based on a sensor design for measuring fertilizer concentrations in a hydroponics or aquaponics operation (Ratcliffe, 2015). The conductivity measurements from the EC sensor are temperature compensated using the temperature data provided by the water temperature sensor. The custom-made EC sensor relies on a simple electrical circuit (DC voltage divider) which can only be used for relatively quick, discrete conductivity measurements (i.e. not for continuous EC measurements). Conductivity measurements with this design can be taken approximately every five seconds (Ratcliffe, 2015). Because this circuit uses DC current rather than AC current, taking conductivity measurements at less than five second intervals may cause the ions in the water to become polarized, leading to inaccurate readings. The custom-made EC sensor was tested against a commercial EC meter (YSI EcoSense pH/EC 1030A) and was found to measure conductivity within approximately 10% of the commercial meter for solutions that are within ± 500 uS/cm of the sensor's calibration value (Appendix A). If desired, the low-cost custom-made EC sensor can be substituted with a commercially probe, such as the Atlas Scientific conductivity probe (<https://atlas-scientific.com/probes/conductivity-probe-k-1-0/>).

The water meter in this report was designed and tested for large diameter (0.9 m inside diameter) dug wells with shallow water depths (less than 10 m below ground surface). However, it could potentially be used for measuring water levels in other situations, such as environmental monitoring wells, drilled wells, and surface water bodies.



Figure 1: External view of meter. Dimensions: 320 mm long (tip to tip), 50 mm diameter. Temperature and EC sensors are not shown; they connect to the meter case with cables.

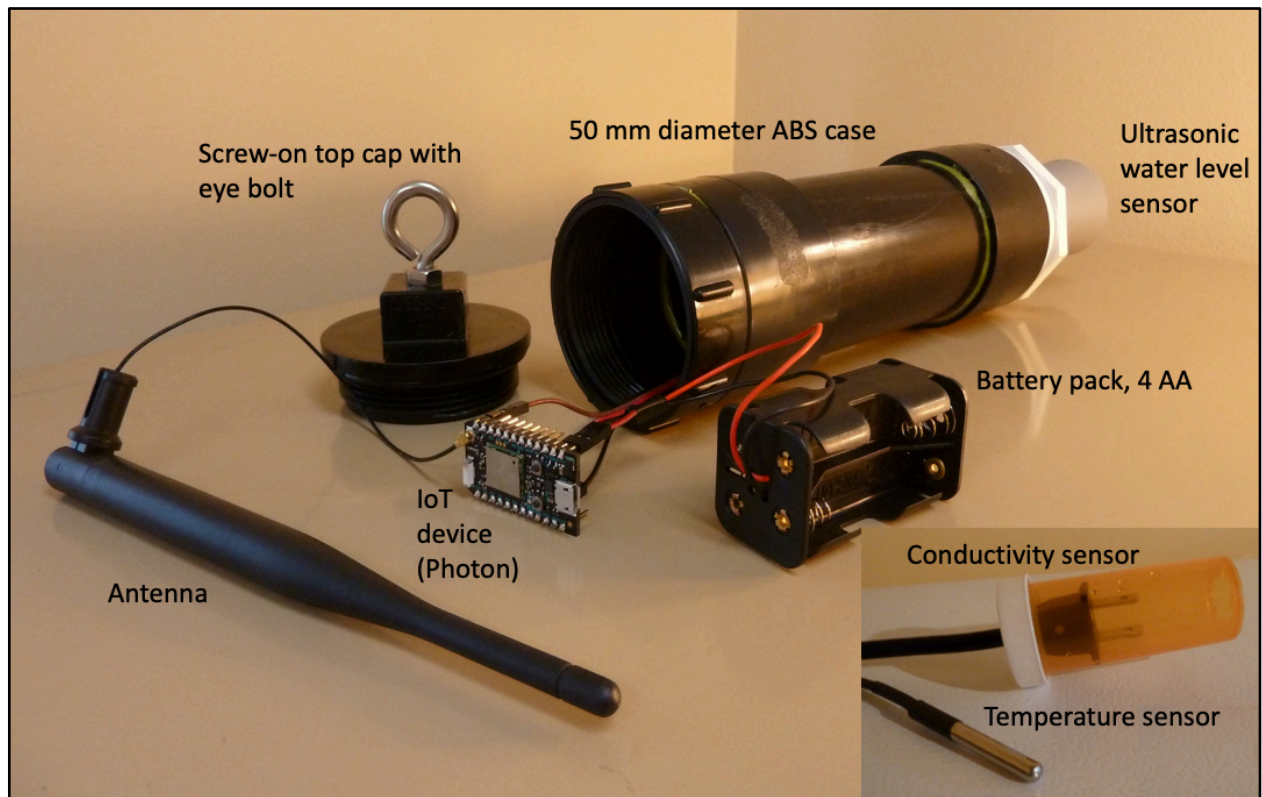


Figure 2: View of meter with top cap removed to show internal parts and sensors.

2.0 Parts List

Table 1 lists the parts needed to build the meter. The electronic parts can be ordered from online retailers such as Amazon. The plumbing and hardware parts for the meter case can be found at building supply stores. The only parts that have been specifically identified with a manufacturer and model number are the ultrasonic sensor (MaxBotix Inc., 2019) and the IoT device (Particle, 2019) because the instructions and code presented here are unique to these parts. However, any similar sensor and IoT device could potentially be substituted for the ones used here, with appropriate assembly and code modifications. The exact choice of wire, antenna, and battery holder is flexible, keeping in mind that the antenna (if the antenna is installed inside the case) and battery holder must fit inside the meter case. Note that the water level sensor listed in Table 1 has a 5 m range. If a 10 m range is needed, it can be substituted with MaxBotix sensor model MB7388. For additional information about using a 10 m range sensor model, see the section below entitled “Meter Construction and Operation Tips”.

Table 1: Parts list.

Electronic parts (WiFi version of the meter)	Cost*
Sensor – MaxBotix MB7389 (5m range) https://www.maxbotix.com/Ultrasonic_Sensors/MB7389.htm	\$132
(Sensor – MaxBotix MB7388 (10m range) https://www.maxbotix.com/Ultrasonic_Sensors/MB7388.htm	
IoT device - Particle Photon with headers https://store.particle.io/collections/wifi/products/photon	\$30
DS18B20 Waterproof Digital Temperature Sensor https://www.robotshop.com/ca/en/ds18b20-waterproof-digital-temperature-sensor.html	\$10
Extension cord for making the conductivity probe – 2 prong, common outdoor cord, 5 m length	\$10
Wire used to extend temperature probe, 4 conductors, 5 m length	\$10
Internal antenna (inside the meter case) – 2.4 GHz, 6dBi, IPEX or u.FL connector, 170 mm long https://www.robotshop.com/ca/en/24ghz-6dbi-antenna-ipex-connector.html	\$7
Large external antenna (only if needed due to poor WiFi signal at well) - 2.4 GHz, 9dBi, with u.FL connector, 390 mm long https://www.amazon.ca/gp/product/B01M3NVW22/ref=ppx_yo_dt_b_asin_title_o03_s00?ie=UTF8&psc=1 and extension cable for external antenna – 3 m length https://www.amazon.ca/gp/product/B001BSK3NO/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&th=1	-
Battery pack – 4 X AA https://www.robotshop.com/ca/en/battery-holder---4-x-aa-compact.html	\$1
Wire – jumper wire with push on connectors (300 mm length) https://www.robotshop.com/ca/en/300mm-40-pin-jumper-wire-splittable.html	\$2
Batteries – 4 X AA	\$5
Plumbing and hardware parts	
Pipe - ABS, 50 mm (2 inch) diameter, 125 mm long	\$1
Top cap, ABS, 50 mm (2 inch), threaded with gasket to make a watertight seal	\$5
Bottom cap, PVC, 50 mm (2 inch) with ¾ inch female NPT thread to fit sensor	\$4
2 Pipe couplers, ABS, 50 mm (2 inch) to connect top and bottom cap to ABS pipe	\$2
Eye bolt and 2 nuts, stainless steel (1/4 inch) to make hanger on the top cap	\$2
Other materials: electrical tape, Teflon tape, pill bottle to make EC sensor cover, solder, silicone, heat shrink, glue to assemble case	\$9
TOTAL =	\$230

*Costs are in Canadian dollars and do not include tax and shipping. Prices are subject to change.

3.0 Meter Construction

Step-by-step instructions for constructing the meter are provided below. It is recommended that the builder read through all constructions steps and the section below entitled “Meter Construction and Operation Tips” before starting the meter construction process. The IoT device used in this project is a Particle Photon and, therefore, in the following sections the terms “IoT device” and “Photon” are used interchangeably.

3.1 Step 1 – Assemble the Meter Case

Assemble the meter case as shown in Figures 1 and 2. The total length of the assembled meter, tip to tip including the water level sensor and eye bolt, is approximately 320 mm. The 50 mm diameter ABS pipe used to make the meter case should be cut to approximately 125 mm in length. This allows sufficient space inside the case to house the IoT device, battery pack, and a 170 mm-long antenna.

Seal all joints with either silicon or ABS glue to make the case watertight. This is very important, otherwise moisture can get inside the case and destroy the internal components. A small desiccant pack can be placed inside the case to absorb moisture.

Install an eye bolt in the top cap of the case by drilling a hole and inserting the eye bolt and nut. A nut should be used on both the inside and the outside of the case to secure the eye bolt. Apply some silicon to the inside of the cap at the bolt hole to make it watertight.

3.2 Step 2 – Attach Wires to Sensors

3.2.1 Water Level Sensor

Three wires (Fig. 3a) must be soldered to the ultrasonic water level sensor in order to attach it to the Photon (i.e. sensor pins GND, V+, and Pin 2). Soldering the wires to the sensor can be challenging because the connection holes on the sensor are small and close together. It is very

important that the wires are properly soldered to the sensor so there is a good, strong physical and electrical connection and no solder arcs between adjacent wires. Good lighting and a magnifying lens help with the soldering process. For those who do not have previous soldering experience, some practice soldering is recommended prior to soldering the wires to the sensor. An online tutorial on how to solder is available from SparkFun Electronics (2020).

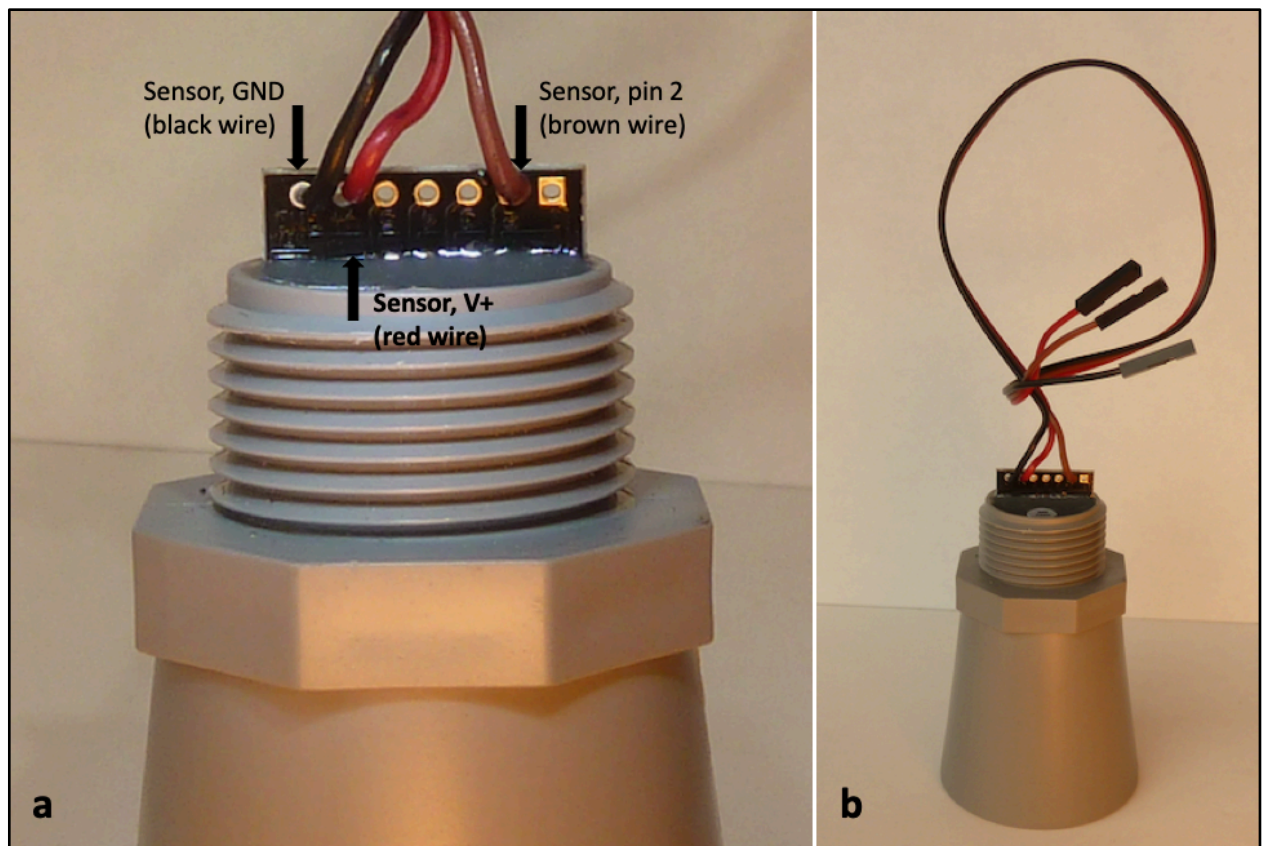


Figure 3: Ultrasonic sensor: a) close-up view showing wire connections; b) full view of sensor showing wires with push-on-type connectors for attaching to the IoT device.

After the wires are soldered to the sensor, any excess bare wire that sticks out from the sensor can be trimmed off with wire cutters to approximately 2 mm length. It is recommended that the solder joints be covered with a thick bead of silicon. This gives the connections more strength and reduces the chance of corrosion and electrical problems at the sensor connections if moisture gets into the meter case. Electrical tape can also be wrapped around the three wires at the sensor

connection to provide additional protection and strain relief, reducing the chance that the wires will break at the solder joints.

The sensor wires can have push-on-type connectors (Fig. 3b) at the one end to attach to the Photon. Using push-on connectors makes it easier to assemble and disassemble the meter. The sensor wires should be at least 270 mm long so they can extend the entire length of the meter case. This length will allow the Photon to be connected from the top end of the case with the sensor in place at the bottom end of the case. Note that this recommended wire length assumes that the ABS pipe used to make the meter case is cut to a length of 125 mm. Confirm in advance of cutting and soldering the wires to the sensor that a wire length of 270 mm is sufficient to extend beyond the top of the meter case so that the Photon can be connected after the case has been assembled and the sensor is permanently attached to the case.

The sensor can now be attached to the meter case. It should be screwed tightly into the bottom cap, using Teflon tape to ensure a watertight seal.

3.2.2 Temperature Sensor

The DS18B20 waterproof temperature sensor has three wires (Fig. 4), which are usually coloured red (V+), black (GND) and yellow (data). These temperature sensors typically come with a relatively short cable, less than 2 m long, which is not long enough to allow the sensor to reach the water level in the well. Therefore, the sensor cable must be extended with a waterproof cable and joined to the sensor cable with a waterproof splice. This can be done by coating the solder connections with silicon, followed by heat shrink. Instructions for making a waterproof splice are provided by Alcox (2016). The extension cable can be made using common outdoor telephone extension line, which has four conductors and is readily available for purchase online at low cost. The cable should be long enough so the temperature sensor can extend from the meter case and be immersed under water in the well, including an allowance for water level drop.



Figure 4: Temperature sensor.

In order for the temperature sensor to work, a resistor must be connected between the red (V+) and yellow (data) wires of the sensor. The resistor can be installed inside the meter case directly on the Photon pins where the temperature sensor wires attach, as listed below in Table 2. The resistor value is flexible. For this project, a 2.2 kOhm resistor was used, however, any value between 2.2 kOhm and 4.7 kOhm will work. The temperature sensor also requires a special code to operate. The temperature sensor code will be added later, as described in Section 3.4 (Software Setup). Further information about connecting a temperature sensor to a Photon can be found in the tutorial here: <https://docs.particle.io/workshops/photon-maker-kit-workshop/ch2/>

The cable for the temperature sensor must be inserted through the meter case so it can attach to the Photon. The cable should be inserted through the bottom of the case by drilling a hole through the case bottom cap (Fig. 5). The same hole can be used to insert the conductivity sensor cable, as described in Section 3.2.3. After the cable is inserted, the hole should be thoroughly sealed with silicon to prevent any moisture from entering the case.



Figure 5: Sensor cables inserted through bottom of case and sealed with silicon.

3.2.3 Conductivity Sensor

The EC sensor used in this project is made from a standard North American Type A, 2 prong electrical plug inserted through a plastic “pill bottle” to control “wall effects” (Fig. 6). Wall effects can affect conductivity readings when the sensor is within about 40 mm of another object. Adding the pill bottle as a protective case around the sensor will control wall effects if the sensor is in close contact with the side of the water well or another object in the well. A hole is drilled through the pill bottle cap to insert the sensor cable and the bottom of the pill bottle is cut off so the water can flow into the bottle and be in direct contact with the plug prongs.

The EC sensor has two wires, including a ground wire and a data wire. It does not matter which plug prong you choose to be the ground and data wires. If a sufficiently long extension cord is used to make the EC sensor, then the cable will be long enough to reach the water level in the well and no waterproof splice will be needed to extend the sensor cable. A resistor must be connected between the data wire of the EC sensor and a Photon pin to provide power. The

resistor can be installed inside the meter case directly on the Photon pins where the EC sensor wires attach, as listed below in Table 2. The resistor value is flexible. For this project, a 1 kOhm resistor was used; however, any value between 500 Ohm and 2.2 kOhm will work. Higher resistor values are better for measuring low conductivity solutions. The code included with these instructions uses a 1 kOhm resistor; if a different resistor is used, the value of the resistor must be adjusted in line 133 of the code.

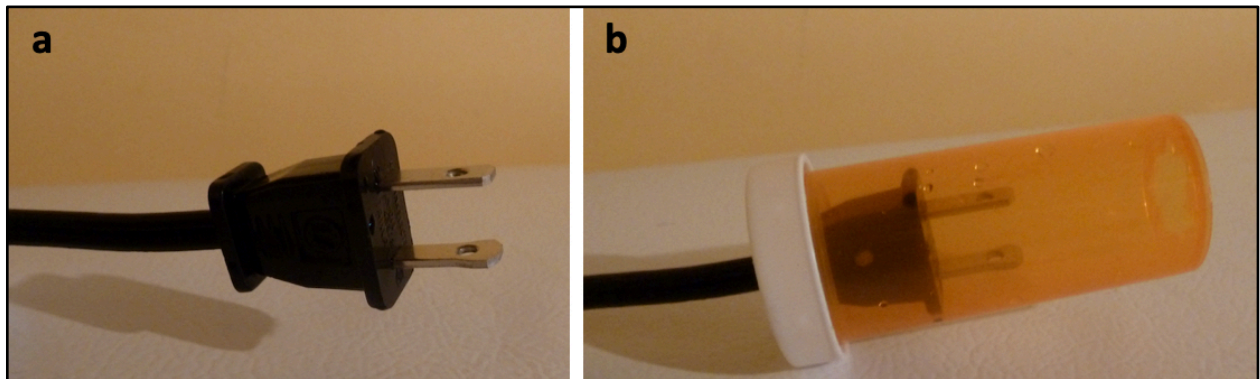


Figure 6: EC sensor made with a common Type A plug: a) plug sensor; b) plug sensor with pill bottle cover to control wall effects.

The cable for the EC sensor must be inserted through the meter case so it can attach to the Photon. The cable should be inserted through the bottom of the case by drilling a hole through the case bottom cap (Fig. 5). The same hole can be used to insert the temperature sensor cable, as described in Section 3.2.2. After the cable is inserted, the hole should be thoroughly sealed with silicon to prevent any moisture from entering the case.

The EC sensor must be calibrated using a commercial EC meter. The calibration procedure is done in the field, as described in Section 5.2 (Field Setup Procedure). The calibration is done to determine the cell constant for the EC meter. The cell constant depends on the properties of the EC sensor, including the type of metal the prongs are made of, the surface area of the prongs, and the distance between the prongs. For a standard Type A plug like the one used in this project, the cell constant is approximately 0.3. Further information on the theory and measurement of

conductivity is available from Radiometer Analytical SAS (2003) and Rosemount Analytical Inc. (2010).

3.3 Step 3 – Attach Sensors, Battery Pack, and Antenna to IoT Device

Attach the three sensors, battery pack, and antenna to the Photon (Fig. 7), and insert all parts into the meter case. Table 2 provides a list of the pin connections indicated in Figure 7. The sensors and battery pack wires can be attached by soldering directly to the Photon or with push-on-type connectors that attach to the header pins on the underside of the Photon (as seen in Fig. 2). Using push-on connectors makes it easier to disassemble the meter or replace the Photon if it fails. The antenna connection on the Photon requires a u.FL type connector (Fig. 7) and needs to be very firmly pushed onto the Photon to make the connection. Do not install the batteries into the battery pack until the meter is ready to be tested or installed in a well. There is no on/off switch included in this design, so the meter is turned on and off by installing and removing the batteries.

Table 2: List of pin connections on the IoT device (Particle Photon).

Photon Pin	Attachment
D2	WL sensor pin 6, V+ (red wire)
D3	WL sensor pin 2, Digital data output (also called the pulse width output) (brown wire)
GND	WL sensor pin 7, GND (black wire)
D5	Temp sensor, data (yellow wire)
D6	Temp sensor, V+ (red wire)
A4	Temp sensor, GND (black wire)
D5 to D6	Temp sensor, resistor R1 (connect a 2.2k resistor between Photon pins D5 and D6)
A0	EC sensor, data
A1	EC sensor, GND
A2 to A0	EC sensor, resistor R2 (connect a 1k resistor between Photon pins A0 and A2)
VIN	Battery pack, V+ (red wire)
GND	Battery pack, GND (black wire)
u.FL	Antenna

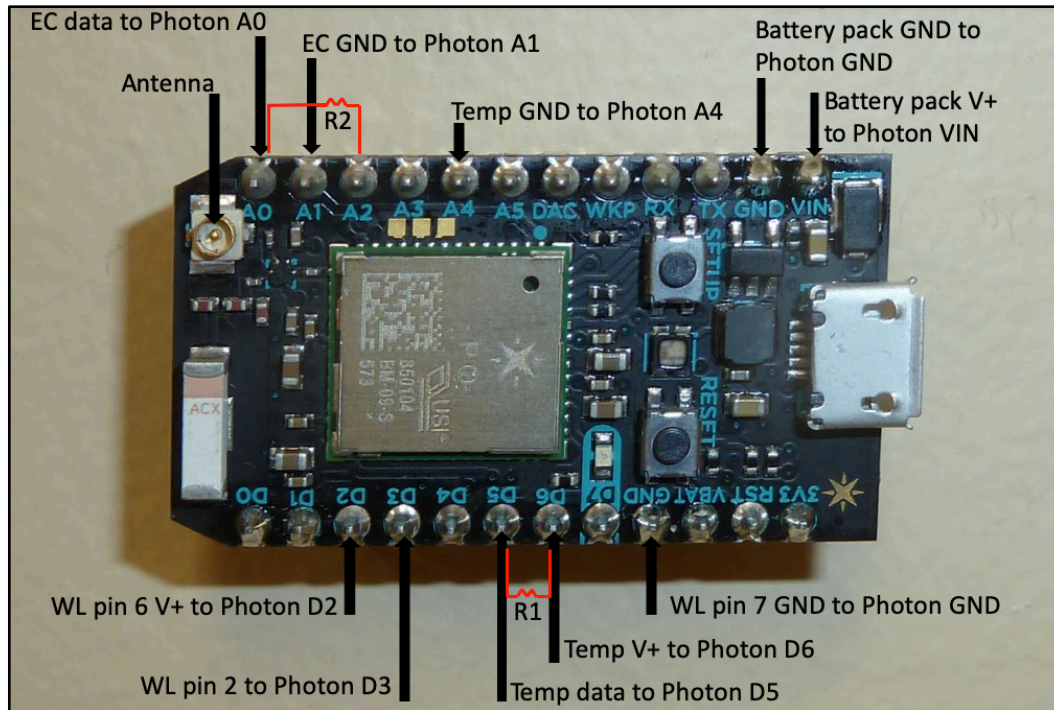


Figure 7: Pin connections on the IoT device (Particle Photon). EC = Electrical Conductivity sensor; Temp = Temperature sensor; WL = Water Level ultrasonic sensor.

3.4 Step 4 – Software Setup

Five main steps are needed to set up the software for the meter:

1. Create a Particle account that will provide an online interface with the Photon. To do this, download the Particle mobile app to a smartphone (<https://docs.particle.io/quickstart/photon/>). After installing the app, create a Particle account and follow the online instructions to add the Photon to the account. Note that any additional Photons can be added to the same account without the need to download the Particle app and create an account again.
2. Create a ThingSpeak account (<https://thingspeak.com/login>) and set up a new channel to display the sensor data. An example of a ThingSpeak webpage is shown in Figure 8. Instructions for setting up a ThingSpeak channel are provided at

<https://docs.particle.io/tutorials/device-cloud/webhooks/>. Note that additional channels for other Photons can be added to the same ThingSpeak account.

3. A “webhook” is required in order to pass the sensor data from the Photon to the ThingSpeak channel. Instructions for setting up a webhook are provided in Appendix B. If more than one meter is being built, a new webhook with a unique name must be created for each additional Photon.
4. Finally, the webhook that was created in the above step must be inserted into the code that operates the Photon. The code for the WiFi version of the water level meter is found at: <https://content.instructables.com/ORIG/FRA/QNKH/KEJYQMTP/FRAQNKHKEJYQMTP.txt>
5. On a computer, go to the Particle webpage (<https://login.particle.io/login?redirect=https://console.particle.io/devices>), login to the Particle account, and navigate to the Particle app interface. Copy the code and use it to create a new app in the Particle app interface. Insert the name of the webhook created above into line 154 of the code. To do this, delete the text inside the quotes and insert the new webhook name inside the quotes in line 154, which reads as follows: "Insert_Webhook_Name_Inside_These_Quotes". The code can now be verified, saved, and installed onto the Photon. When the code is verified it will return an error that says “OneWire.h: No such file or directory”. OneWire is the library code that runs the temperature sensor. This error must be fixed by installing the OneWire code from the Particle library. To do this, go to the Particle App interface with your code displayed and scroll down to the Libraries icon on the left-hand side of the screen (located just above the question mark icon). Click on the Libraries icon, and search for OneWire. Select OneWire and click “Include in Project”. Choose the name of your app from the list, click “Confirm” and then save the app. This will add three new lines to the top of the code. These three new lines can be deleted without affecting the code. It is recommended that you delete these three lines so that the code line numbers will match the instructions in this document. If the three lines are left in place, then all code line numbers discussed in this document will be advanced by three lines.

Note that the code is stored in and installed onto the Photon from the cloud. This code will be used to operate the meter when it is in the water well. During the field installation, some changes will need to be made to the code to set the reporting frequency to once a day and add information about the water well, as described below in the section entitled “Installing the Meter in a Water Well”.

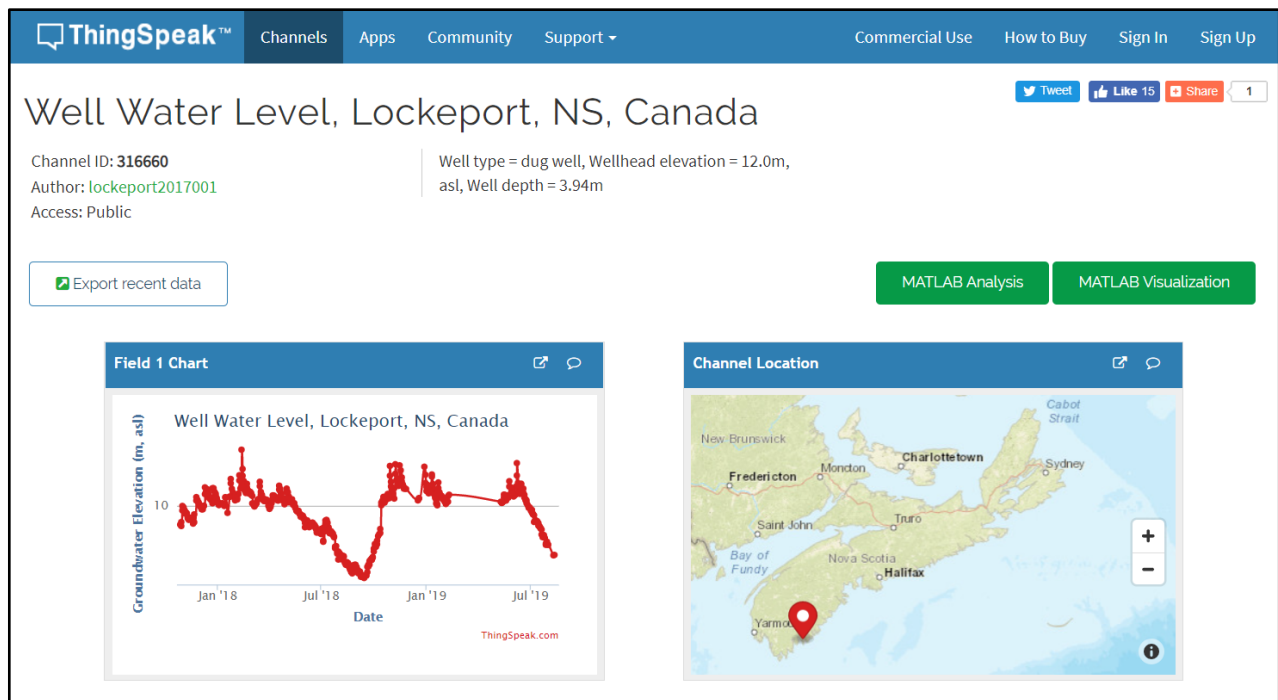


Figure 8: Webpage for a dug well located in Lockeport, Nova Scotia, Canada
(<https://thingspeak.com/channels/316660>).

3.5 Step 5 – Test the Meter

The meter construction and software setup are now complete. At this point it is recommended that the meter be tested. Two tests should be completed. The first test is used to confirm that the meter can correctly measure water levels, EC values and temperature and send the data to ThingSpeak. The second test is used to confirm that the power consumption of the Photon is within the expected range. This second test is useful because the batteries will fail sooner than expected if the Photon is using too much power.

For testing purposes, the code is set to measure and report the sensor data every two minutes. This is a practical time period to wait between measurements while the meter is being tested. If a different measurement frequency is desired, change the variable called MeasureTime in line 19 of the code to the desired measurement frequency. The measurement frequency is entered in seconds (i.e. 120 seconds equals two minutes).

The first test is done to confirm the sensors are working properly. This can be done in the office by hanging the meter above the floor, turning it on, and checking that the ThingSpeak channel accurately reports the distance between the sensor and the floor. In this testing scenario the ultrasonic pulse reflects off the floor, which is used to simulate the water surface in the well. The EC and temperature sensors can be placed in a container of water of known temperature and conductivity (i.e. as measured by a commercial EC meter) to confirm the sensors report the correct values to the ThingSpeak channel.

For the second test, the electrical current between the battery pack and the Photon should be measured to confirm that it matches the specifications in the Photon datasheet (Particle, 2019). Experience has shown that this test helps identify defective IoT devices before they are deployed in the field. Measure the current by placing a current meter between the positive V+ wire (red wire) on the battery pack and the VIN pin on the Photon. The current should be measured in both operating mode and deep sleep mode. To do this, turn the Photon on and it will start up in operating mode (as indicated by the LED on the Photon turning a cyan colour), which runs for approximately 20 seconds. Use the current meter to observe the operating current during this time. The Photon will then automatically go into deep sleep mode for two minutes (as indicated by the LED on the Photon turning off). Use the current meter to observe the deep sleep current at this time. The operating current should be between 80 and 100 mA, and the deep sleep current should be between 80 and 100 μ A. If the current is higher than these values, the Photon should be replaced.

4.0 Meter Construction and Operation Tips

It is very important that the meter case is watertight. The IoT device can be destroyed if any moisture enters the case. As such, it is important to seal any holes and ABS pipe joints with silicon or glue. If an external antenna is added to the meter, it is recommended that the antenna cable be inserted through the bottom of the case and thoroughly sealed with silicon to prevent moisture entry (Fig. 9).



Figure 9: External antenna cable inserted through case bottom and sealed with silicon.

The MaxBotix sensor listed in Table 1 (model MB7389) has a 5 m range. If a 10 m range is needed, use the MaxBotix sensor model MB7388. The software code presented in these instructions applies to the 5 m range model. When using the 10 m range model, please adjust the code by changing the number 0.31 to 0.51 in line 142 of the code and 4.99 to 9.99 in line 146 of

the code. This adjusts the code to the range provided by the 10 m sensor model by allowing only water depths between 0.5 m and 10 m to be reported.

The field installation instructions provided in the section below describe how to set the code to take measurements once a day, but a different measurement frequency can be used by changing the variable called MeasureTime in line 19 of the code to the desired measurement frequency. The measurement frequency is entered in seconds (i.e. 120 seconds equals two minutes and 86400 seconds equals one day). Changing the measurement frequency will affect the battery life of the meter. The battery life is expected to be approximately 1 to 2 years with a measurement frequency of once a day. At the time of writing, the longest observed battery life in the field has been 2 years. Using a higher measurement frequency will reduce the battery life.

The EC sensor's temperature compensation uses a linear temperature coefficient of 0.02 per degree centigrade (i.e. 2%), which is commonly used for drinking water applications (Radiometer Analytical SAS, 2003). The user can adjust this temperature coefficient by modifying line 136 of the code.

The conductivity measurement from the EC sensor is used by the code to calculate the Total Dissolved Solids (TDS) concentration in milligrams per Litre (mg/L) of the water. The calculation is done using a linear equation with a slope factor of 0.59, as discussed by Hem (1985). The user can adjust this slope factor by modifying line 137 of the code.

5.0 Installing the Meter in a Water Well

5.1 Before Going to the Field

Check the local regulatory requirements regarding applicable qualifications and procedures for opening a water well (such as well disinfection requirements after removing a well cap) and take precautions not to introduce contamination into the well while installing the well meter.

Prior to going into the field, and while the Photon is still turned on and connected to a WiFi network, install the Tinker app from the Particle interface onto the Photon. The Tinker app will keep the Photon awake so a connection to the well owner's WiFi network can be made. The meter's operating code cannot be used to make the initial WiFi connection because, as an energy-saving feature, it is programmed to shutoff the Photon if no WiFi signal is found within 40 seconds. This limit of 40 seconds is not enough time to complete the initial WiFi configuration in the field. After the WiFi is configured on the Photon, the meter's operating code can be re-installed.

To install the Tinker app, use a computer to go to the Particle webpage (<https://login.particle.io/login?redirect=https://console.particle.io/devices>), login to the Particle account, and navigate to the Particle app interface (Particle Web IDE). The Tinker app is available at the bottom of the webpage under example apps. Select the Tinker app and install it onto the Photon. The Tinker app can also be installed in the field without using the Internet by connecting the Photon directly to a computer with a USB connection. This is done by placing the Photon in DFU mode and using the Particle CLI program to install the Tinker app over the USB connection. For instructions, please see the section on "Firmware Reset" at <https://docs.particle.io/tutorials/device-os/led/photon/>.

5.2 Field Setup Procedure

Upon arrival at the well, check that a smartphone or computer can successfully connect to the well owner's WiFi network while standing at the wellhead. Please see the section below entitled "Field Installation Tips" if the smartphone is unable to connect to the WiFi network at the well.

Open the water well and use a manual water level tape (or common carpenter's tape measure) to measure the depth to water from the top of the well casing. This will be used to calculate an offset value that will be added to the code so it will correctly report the depth to the water. Use a commercial EC meter to measure the conductivity of the well water. This will be used to calibrate the meter's EC sensor. If the well is being used as a water supply, the conductivity of

the well water can be measured by running water from a tap connected to the well's water system until the conductivity has stabilized. The conductivity measured by the commercial EC meter must be entered in the meter's code in line 26 for the variable named "calibration". Save the code after the conductivity value has been entered into the code.

Turn the meter on by installing batteries in the battery pack and attaching the battery pack to the Photon. The Photon should now be running the Tinker app, as discussed in the section above. Use the Particle mobile app on a smartphone to add a new device to the Particle account using the well owner's WiFi network name and password. Once the Photon is successfully connected to the WiFi network, insert the Photon into the meter case and screw the top cap tightly onto the case to make a watertight seal.

Hang the meter in the well (Fig. 10) and close the well cover. A hanger may need to be installed inside the well to hang the meter. Next, use the Particle App interface on a laptop computer to install the meter's operating code onto the Photon. The code that should be used is the version that was created in Step 4 – Software Setup in Section 3.4 with the new webhook name inserted into line 154 and the "calibration" value updated with the measured conductivity of the well water, as described above. This code has a two-minute reporting frequency, which is temporarily used to complete the meter setup. The code will be set to a frequency of once a day at the end of the setup process. Confirm that the meter is successfully connecting to the Internet while inside the water well and reporting the sensor data to the ThingSpeak channel every two minutes.



Figure 10: Meter installed in a dug well.

The following paragraphs describe how to enter user-specified, site-specific variables into the code, which are determined by the user for each water well. These variables include the meter “HangDown” (the distance between the well datum and the meter’s water level sensor), the well “Datum” (the datum used to convert the water level measured by the meter to a water elevation, relative to mean sea level), “MeasureTime” (the frequency that the meter will take measurements), “DelayTime” (used to delay the measurements so they will occur at a time of day chosen by the user) and the EC sensor’s cell constant “K” (used to calibrate the EC sensor to the known conductivity of the well water, as measured by a commercial EC meter).

While the meter is still hanging inside the well and taking measurements every two minutes, use the water level reported by the meter on ThingSpeak to calculate the meter hang-down distance between the well datum and the meter’s sensor and enter this value into line 23 of the code for the variable named “HangDown” (entered in metres). This value is used to adjust the meter reading so it reports the distance from the datum to the water level. To calculate the HangDown value, subtract the water level reported by the meter from the water level measured manually earlier. For example, if the water level measured by water level tape is 3.0 m below the top of the casing and the meter reports the distance to water as 2.5 m, then the HangDown value is 0.5 m (i.e. $3.0\text{ m} - 2.5\text{ m} = 0.5\text{ m}$).

Enter the elevation of the well datum (i.e. top of casing elevation) with respect to mean sea level in line 22 of the code for the variable named “Datum” (entered in metres). This is used to convert the depth to water measurement to a groundwater level elevation.

Update the EC sensor’s cell constant K in line 25 for the variable named “K”. The correct K value is automatically calculated by the meter’s code, but needs to be manually inserted into the code by the user. To obtain the correct K value, wait until the meter turns on and comes online, then click on the Photon device within the Particle App interface. This will display the details of the Photon device. On the bottom right hand side of the screen click on “Variables” and then click on the “Get” button beside the variable named “CalculatedK”. Enter this value into line 25 for the variable named “K” (five decimal places should be sufficient). The K value depends on the conductivity of the water and properties of the EC sensor, but it should be approximately 0.3.

Change the monitoring frequency in the code to once a day by changing the “MeasureTime” (entered in seconds) from 120 to 86400 seconds in line 19 of the code.

If desired, set the “DelayTime” (entered in seconds) in line 20 of the code to delay the measurements to a time of day chosen by the user. This feature is used if the user wants the water measurements to be taken at a specific time of day. For example, in the middle of the night when the well is not being pumped and the water level in the well has recovered to its static level. If this feature is not used, then the meter will take daily measurements at the same time each day, starting when the code is installed on Photon. In order to activate this feature, the user must first run the code with a 2 minute measurement frequency, which allows line 90 of the code to set the Counter to zero; the user then must place two backslashes in front of line 90 of the code to prevent line 90 from running (`//Counter=0;`) and flash the code one final time. The batteries must remain connected to the Photon when doing this so the Counter value of zero is retained in the memory. When activated, the first measurement is taken immediately when the code is installed on the Photon and the next measurement, taken on the following day, will be delayed by the specified DelayTime. All future measurements will be taken at the same time as the

measurement on the second day. If a delay time is not needed, do not enter a DelayTime and do not insert backslashes in front of line 90 of the code.

After all of the above user-specified variables have been entered, verify and save the code. The final code is now ready to be installed onto the Photon so that it will take water measurements once a day and report the data to ThingSpeak. At this point the Photon is still turning on and taking measurements every two minutes. The next time the Photon turns on and comes online, install the newly updated code onto the Photon. After installing the updated code, sometimes the 2-minute measurement frequency code will run one more time before the updated code takes effect. The installation process is now complete.

5.3 Field Installation Tips

A common problem in the field is a weak WiFi signal that prevents the meter from connecting to the Internet. The WiFi signal strength (RSSI) should be checked at the wellhead on a smartphone or computer. The Photon also reports WiFi signal strength as part of its online diagnostic tools. Signal strengths are presented as negative values and the closer to zero, the stronger the signal. For example, -60 dBm is a stronger signal strength than -70 dBm. WiFi signal strengths of less than -80 dBm (i.e. -90 or -100 dBm) can be unreliable. In these cases, a WiFi repeater can be added to boost the WiFi signal strength and/or an external antenna can be attached to the meter. The external antenna can be connected with a long (3 m) cable that allows the antenna to be attached to the outside of the well at the wellhead (Fig. 11). It is recommended that the antenna cable be inserted through the bottom of the case and thoroughly sealed with silicon to prevent moisture entry (Fig. 9). A good-quality, waterproof, outdoor coaxial extension cable is recommended.

In cold climates, condensation and frost can form on the water level sensor face when outdoor air temperatures drop below freezing. This can block the ultrasonic beam and cause inaccurate water level readings. The problem can usually be solved by lowering the meter into the well by 1 to 2 m where temperatures are warmer. This approach has been tested in the field and found to

solve most condensation and frost problems. An external antenna must be added in this case because the WiFi signal will likely be too weak when the meter is lowered deeper into the well.



Figure 11: External antenna on side of well with cable leading to meter inside well.

Field testing has shown that the meter works well in large diameter dug wells with smooth interior walls (i.e. concrete or plastic casing). However, problems with inaccurate water level readings have been encountered in older dug wells that are constructed with rocks instead of smooth-walled concrete or plastic casing. The presence of irregularly shaped rocks inside the well may cause the ultrasonic beam to reflect off the rocks rather than the water surface. This can be corrected by placing a small-diameter (75 mm or greater) PVC sleeve in the well that extends from the wellhead to below the water level and placing the water meter inside the PVC sleeve. This prevents the rocks from influencing the ultrasonic beam. It should be noted that the MaxBotix sensor model used here is specifically designed to ignore small acoustic targets and report the distance to the most likely target (i.e. the water surface). However, if there are significant obstacles in the well above the water level, such as pipes and wires, these obstacles may interfere with the ultrasonic beam and cause inaccurate water level readings, like the rocks discussed above. A PVC sleeve can also be installed in the well to solve this problem.

6.0 How to Make a Cellular Version of the Meter

A cellular version of the meter can be built by making modifications to the previously described parts list, instructions and code. The cellular version does not require WiFi because it connects to the Internet via a cellular signal. The cost of the parts to build the cellular version of the meter is approximately Can\$330 (excluding taxes and shipping), plus approximately Can\$4 per month for the cellular data plan that comes with the cellular IoT device.

The cellular meter uses the same parts listed in Table 1 and construction steps listed above with the following modifications:

- Substitute the WiFi IoT device (Particle Photon) for a cellular IoT device (Particle Electron) (<https://store.particle.io/collections/cellular/products/electron-3g-americas>). The cost for this cellular IoT device is approximately Can\$110 for a 3G version with a 3.7 V Li-Po battery included. When constructing the meter, use the same pin connections described above for the WiFi meter in Step 3 (Section 3.3).
- The cellular IoT device uses more power than the WiFi version and, therefore, two battery sources are recommended: a 3.7V Li-Po battery, which comes with the IoT device, and the battery pack (4 AA batteries) listed in Table 1. The 3.7V LiPo battery attaches directly to the IoT device with the connectors provided. The AA battery pack is attached to the IoT device the same way as described above for the WiFi meter in Step 3 (Section 3.3). Field testing has shown that the cellular version of the meter will operate for approximately 9 months using the battery setup described above. An alternative to using both the AA battery pack and 2000 mAh 3.7 V Li-Po battery is to use one 3.7V Li-Po battery with a higher capacity (e.g. 4000 or 5000 mAh).
- An external antenna must be attached to the meter, such as: (https://www.amazon.ca/gp/product/B07PZFV9NK/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&psc=1). Ensure it is rated for the frequency used by the cellular service provider where the meter will be used. The antenna that comes with the cellular IoT device is not suitable for outdoor use. The external antenna can be connected with a long (3 m) cable that allows the antenna to be attached to the outside of the well at the

wellhead (Fig. 11). It is recommended that the antenna cable be inserted through the bottom of the case and thoroughly sealed with silicon to prevent moisture entry (Fig. 9). A good-quality, waterproof, outdoor coaxial extension cable is recommended.

- The cellular IoT device runs on a different code than the WiFi version of the meter. The code for the cellular version of the meter is found at: (<https://content.instructables.com/ORIG/F5I/X150/KEJYQNDR/F5IX150KEJYQNDR.txt>)

8.0 References

Alcox, D., 2016. Creating and weatherproofing a wire splice; MaxBotix Inc. <<https://www.maxbotix.com/Tutorials/133.htm>> [Accessed July 28, 2020]

Drage, J., 2020. A real-time well water level meter; Instructables Circuits, Autodesk Inc. <<https://www.instructables.com/id/A-Real-Time-Well-Water-Level-Meter/>> [Accessed July 28, 2020]

Hem, J.D., 1985. Study and interpretation of the chemical characteristics of natural water, 3rd edition; U.S. Geological Survey Water-Supply Paper 2254 <<https://pubs.usgs.gov/wsp/wsp2254/pdf/wsp2254a.pdf>>

MaxBotix Inc., 2019. HRXL-MaxSonar®-WR™ series; MaxBotix Inc., 22 p. <https://www.maxbotix.com/documents/HRXL-MaxSonar-WR_Datasheet.pdf> [Accessed June 3, 2019]

Ousley, T., 2015. Wi-Fi twitter water level sensor; Instructables Circuits, Autodesk Inc. <<https://www.instructables.com/id/Wi-Fi-Twitter-Water-Level-Sensor/>> [Accessed October 20, 2016]

Particle, 2019. Photon datasheet (V016), model number: PHOTONH, PHOTONNOH; Particle <<https://docs.particle.io/datasheets/wi-fi/photon-datasheet/>> [Accessed June 12, 2019]

Radiometer Analytical SAS, 2003. Conductivity theory and practice. <https://support.hach.com/ci/okcsFattach/get/1002532_4> [Accessed August 25, 2020]

Ratcliffe, M., 2015. Three dollar EC - PPM meter [Arduino]; Hackaday <<https://hackaday.io/project/7008-fly-wars-a-hackers-solution-to-world-hunger/log/24646-three-dollar-ec-ppm-meter-arduino>> [Accessed July 28, 2020]

Rosemount Analytical Inc., 2010. Theory and application of conductivity.

<<https://www.emerson.com/documents/automation/application-data-sheet-theory-application-of-conductivity-rosemount-en-68442.pdf>>

[Accessed July 28, 2020]

SparkFun Electronics, 2020. How to solder: through-hole soldering; SparkFun Electronics

<<https://learn.sparkfun.com/tutorials/how-to-solder-through-hole-soldering>> [Accessed July 7,

2020]

ThingSpeak, 2019. ThingSpeak for IoT projects: Data collection in the cloud with advanced data

analysis using MATLAB; The MathWorks Inc. <<https://thingspeak.com/>> [Accessed June 3,

2019]

Appendix A – Conductivity Sensor Verification

Two verification tests were carried out on two different EC sensors: 1) an EC sensor made from a common Type A electrical plug, and 2) a commercial EC sensor made by Atlas Scientific (Model# ENV-40-EC-K1.0; <https://atlas-scientific.com/probes/conductivity-probe-k-1-0/>).

The first test was a bench top test in which the two EC sensors were calibrated to a solution of known conductivity (of approximately 500 uS/cm) and then tested with several different solutions with conductivities ranging from approximately 100 uS/cm to 1000 uS/cm. The results were compared to measurements made with a commercial conductivity meter (YSI EcoSense pH/EC 1030A). Temperature compensation was used with the two EC sensors and the commercial meter.

The second test was completed in the field with the real-time meter installed in a water well taking daily measurements for 82 days. For the field test, both EC sensors were connected to the same meter and were tested simultaneously. Verification measurements were made with a commercial conductivity meter (YSI EcoSense pH/EC 1030A). The verification samples were collected periodically from a kitchen tap sourced by the water well. The tap was allowed to run until EC measurements stabilized.

The results are provided in Tables A.1 and A.2. The results indicate that the two EC sensors performed similarly. Both had mean absolute errors between 4% and 6% and maximum errors of approximately 10%, compared to the commercial EC meter. The errors of both EC sensors increased as the difference between conductivity of the calibration and test solution increased.

Table A.1 Bench Top Verification Data

Test Solution EC (uS/cm) YSI EcoSense Meter	Plug EC Sensor			Atlas EC Sensor		
	EC (uS/cm)	Error (uS/cm)	Error (%)	EC (uS/cm)	Error (uS/cm)	Error (%)
972	900	-72	-8%	893	-79	-9%
517*	517	0	0%	516	-1	0%
293	298	14	5%	298	5	2%
83	93	10	11%	86	3	3%
Mean absolute error =			6%			
				4%		

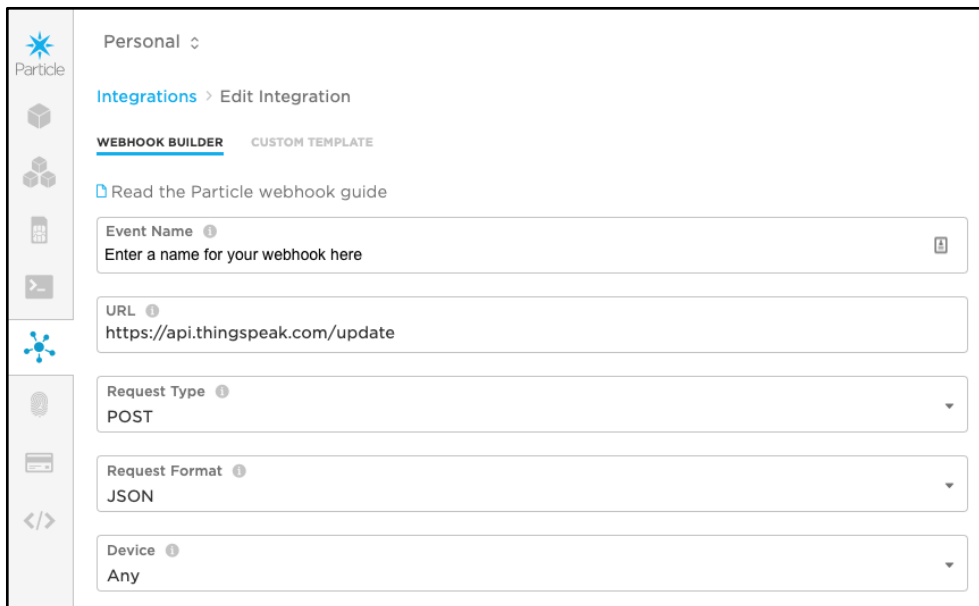
* Both EC sensors were calibrated with the 517 uS/cm solution.

Table A.2 Field Verification Data

Date	EC (uS/cm) YSI EcoSense Meter	Plug EC Sensor			Atlas EC Sensor		
		EC (uS/cm)	Error (uS/cm)	Error (%)	EC (uS/cm)	Error (uS/cm)	Error (%)
27-May-2020	97	97	0	0%	99	2	2%
09-Jun-2020	114	104	-10	-9%	111	-3	-3%
24-Jun-2020	112	104	-8	-7%	106	-6	-5%
30-Jun-2020	109	105	-4	-4%	105	-4	-4%
11-Jul-2020	113	104	-9	-8%	103	-10	-9%
02-Aug-2020	103	101	-2	-2%	99	-4	-4%
11-Aug-2020	101	101	0	0%	100	-1	-1%
17-Aug-2020	105	101	-4	-4%	99	-6	-6%
Mean absolute error =				4%			
					4%		

Appendix B - Webhook Setup Instructions

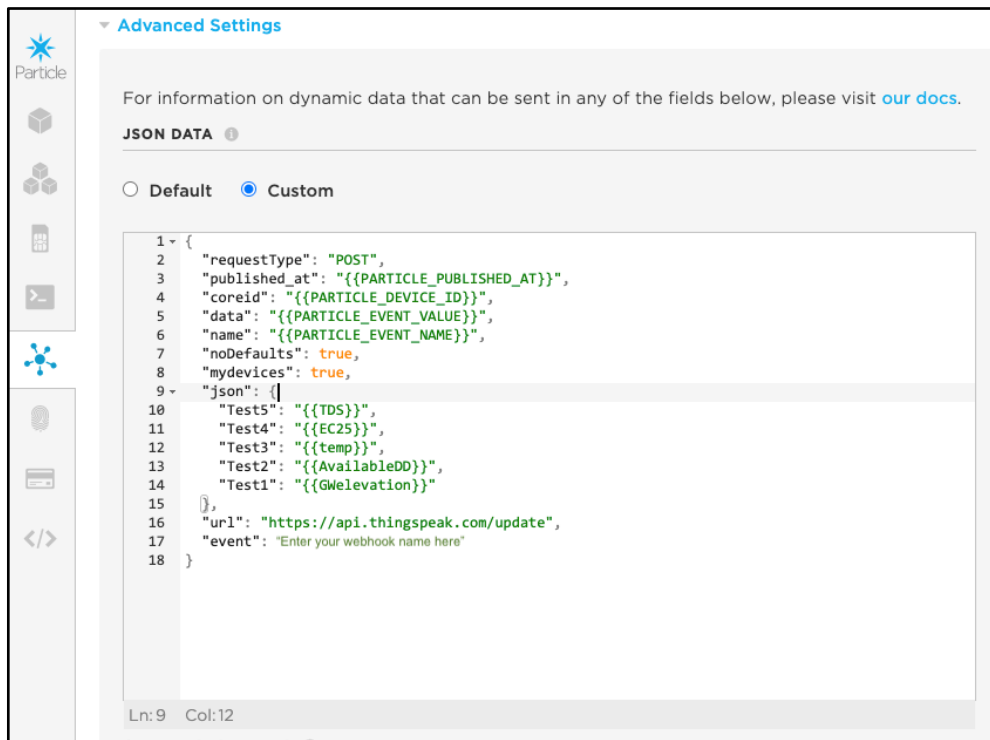
1. Using a desktop/laptop computer, go to the Particle webpage and login to your account and click on Console: <https://login.particle.io/login?redirect=https://console.particle.io>
2. On the left side of the screen, click on Integrations (<https://console.particle.io/integrations>), New Integration, and select Webhook.
3. Fill in the blank fields as shown in the screenshot below. All fields should be exactly as shown below, except the Event Name field, in which you type a name for your unique webhook. Make sure to enter JSON in the Request Format field (not Web Form).



The screenshot displays the Particle Webhook Builder interface. On the left is a sidebar with the Particle logo and navigation icons. The main content area shows the 'Personal' user profile and 'Integrations > Edit Integration' page. Two tabs are visible: 'WEBHOOK BUILDER' (active) and 'CUSTOM TEMPLATE'. A link to 'Read the Particle webhook guide' is present. Below are five input fields:

- Event Name:** A text input field with a placeholder 'Enter a name for your webhook here' and an information icon.
- URL:** A text input field containing 'https://api.thingspeak.com/update'.
- Request Type:** A dropdown menu with 'POST' selected.
- Request Format:** A dropdown menu with 'JSON' selected.
- Device:** A dropdown menu with 'Any' selected.

4. Under Advanced Settings, select the Custom option and fill in the fields as shown in the screenshot below. To do this, you can cut and paste the code shown below. Make sure to enter the name of your webhook name for the “event” in code line 17.



```

{
  "requestType": "POST",
  "published_at": "{{PARTICLE_PUBLISHED_AT}}",
  "coreid": "{{PARTICLE_DEVICE_ID}}",
  "data": "{{PARTICLE_EVENT_VALUE}}",
  "name": "{{PARTICLE_EVENT_NAME}}",
  "noDefaults": true,
  "mydevices": true,
  "json": {
    "Test5": "{{TDS}}",
    "Test4": "{{EC25}}",
    "Test3": "{{temp}}",
    "Test2": "{{AvailableDD}}",
    "Test1": "{{GWelevation}}"
  },
  "url": "https://api.thingspeak.com/update",
  "event": "Enter your webhook name here"
}

```

- Continue to fill in the next fields, as shown in the following screenshot. Note that you will need to go to your ThingSpeak.com (<https://thingspeak.com/login>) account and channel that you plan to use, and copy your Write API Key to use in the webhook, as shown in the screenshot below.

Field Name	ThingSpeak Field ID
api_key	Insert your ThingSpeak API Key here
field1	{{GWelevation}}
field2	{{AvailableDD}}
field3	{{temp}}
field4	{{EC25}}
field5	{{TDS}}

+ ADD ROW

Channel ID: 112

Author: jme783
Access: Private

Private View Public View Channel Settings **API Keys** Data Import / Export

Write API Key

Key: S2EHZVZYE04RTLKA

[Generate New Write API Key](#)

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- Write API Key: Use this key to write data to a channel. If you feel your key has been compromised, click Generate New Write API

- Continue to scroll down and complete the rest of the fields as shown in the screenshots below. Then click Create Webhook and you are finished.

Particle

HTTP BASIC AUTH

Username

Password

HTTP HEADERS

Content-Type > application/json

Accept > application/json

> > ×

+ ADD ROW

Particle

> > ×

+ ADD ROW

WEBHOOK RESPONSES

Response Topic

Error Response Topic

Response Template

ENFORCE SSL

Yes No

CREATE WEBHOOK