Low Water Crossing Indicator

FINAL REPORT

Team Number: sddec22-10
Client: Lee Harker
Advisers: Lee Harker
Team Members/Roles:
Tyler Rebischke: Team Lead/Solar
Dylan Blattner: Product Owner/Sensors
Brandon Choy: Communications
Jacob Ross: Power Systems
Nithin Sebastian: Signage/Alerting
Team Email: sddec22-10@iastate.edu
Team Website: https://sddec22-10.sd.ece.iastate.edu

Revised: December 2022- Version 1

Table of Contents:

Problem Statement:

In thousands of places in the country including a few hundred in Iowa, there are minimum service roads interrupted by small creeks or rivers. On many of these roads, a low water crossing is created at a fraction of the cost of an actual bridge. These roads are used by farmers and residents in the area on a basis limited by the recent rainfall and drainage. When the flow is low, a regular vehicle can drive through the stream. When the flow is higher, it may only be crossed by a high clearance vehicle or tractor and sometimes it is impassable. If the water is too high, the driver would need to make a u-turn on a generally narrow minimally maintained road. Other than signs warning of possible high water, there is no way for the driver to know the actual condition except by driving within a few yards where they can see the water. Drivers usually have an option to take an alternate route but it may be many miles longer. The goal of this project would be to create a water level device and one or more remote annunciation devices so that drivers could see the level of the upcoming low water crossing before driving the length of the road and then possibly needing to turn around and backtrack.

Project Goal:

Our goal for this project was to create a proof of concept prototype low water level indicator to prove that this device is feasible and effective.

Revised project design:

How our design has evolved since EE/CPRE/SE 491:

Our project has evolved in several key ways since our first semester of senior design. The main evolution has come in how we are displaying the water level information to drivers. In 491, we originally planned on having some sort of ruler-shaped sign with LED lights that displayed the depth level to drivers continuously all day every day. After doing extensive research into stream levels, driver distraction studies, and testing, we decided to instead change our design to only have the LEDs on if the water level has hit a depth that begins to become dangerous for drivers. We made this decision for several reasons. First, by only having the lights on when water levels are dangerous we were able to massively cut the power requirements of our system, which means that we could reduce solar panel and battery size, which also ended up reducing the costs for implementation quite significantly. Second, we found through some testing that it makes much more sense to only alert drivers when conditions are hazardous compared to telling them the raw depth due to the fact that most people have no idea what depth is actually safe for their cars to drive through. Our final design flashes yellow when it is unsafe for smaller vehicles like sedans to use the low water crossing and flashes red when it is unsafe for larger vehicles like SUVs and pickup trucks to cross.

Functional Requirements:

Our project consisted of the following functional requirements:

- Our solution needs to accurately be able to measure water levels while consuming little power
- Due to the remote location in which our product is going to be deployed, we need to have our product be solar powered.
- Also due to the remote location of our solution, it needs to be very low maintenance with visits from maintenance crews being required only once every several months.
- As our product will be outdoors during all four seasons all day, it needs to be robust and weatherproof enough to handle a diverse set of conditions; it will be exposed to both roadside weather conditions and streamside weather and water conditions.
- Our product needs to be dependable with limited or no downtime. It also needs to be sturdy enough to handle high winds and water flow.
- Our solution needs to be able to accurately display water levels to drivers during both the day and nighttime, while also not distracting drivers from the road.

Non-functional Requirements:

The non-functional requirements for our project fall into three main categories: environmental, aesthetic, and economic.

Environmental:

- Our project should have a low environmental impact on the stream and surrounding habitat.
- Our project will use renewable energy to power the systems.

Aesthetics:

- Our product should be contained in an aesthetically pleasing box that both protects the electronic components and provides waterproofing.
- The LED display/signage should be aesthetically pleasing and easy to understand.

Economic

- Stay within $500 budget if possible

Relevant Standards:

IEEE 802.15.4-2003. This standard is relevant because it lays out guidelines for low power low data rate RF Communications. We plan on using low power and low data rate communications so this standard will help us to develop our strategy.

IEEE /ISO/IEC 17464-2021. This standard is relevant because it lays out guidance for software life cycle processes and maintenance. Because we will be using some open source software for our project we need to be comfortable with maintaining that software along with the software that we write.

IP56/IP64. This standard pertains to weatherproofing for electronics components that are to be used outside. This standard is extremely important to us as all of our components are to be kept outside for four seasons. We will be able to design the housings for our devices based on the requirements laid out in these standards.

ASTM (American Society for Testing and Materials) D4956(Type 1). This specification covers flexible, non-exposed glass bead lens and microprismatic, retroreflective sheeting designed for use on traffic control signs, delineators, barricades, and other devices.

Engineering Constraints:

- Quality: Our product needed to be constructed out of sturdy, dependable materials that would remain weatherproof in any conditions. Quality would also help to reduce the long term maintenance requirements of the project which would help to make it more viable in the long run.
- Accessibility: Our project had a huge accessibility requirement as it needed to be very easily readable and interpretable by people from all walks of life and all levels of education. This constrained how technical we wanted to make the sign. As we had mentioned we originally wanted to have a ruler type of sign, but opted to go with a more easy to understand sign with only flashing lights due to this constraint.
- Power: Our project also had a substantial power constraint. We needed to ensure that we were able to power all of our systems dependably while also keeping our power stores small and manageable.
- Cost: We had an initial cost constraint of $500 with a bit of flexibility built in which we took full advantage of.
- Time: Our project had a pretty serious time constraint. Our team members needed to manage a full load of classwork while also committing to designing and innovating our design. This constraint led to a lack of iterations of our project.

Security Concerns and Countermeasures

Physical:

The primary security concerns come from physical threats. The main concern is that someone would gain physical access to our waterproof boxes and either steal our parts or sabotage them. The primary countermeasure we came up with would be to recommend that customers apply proper locks to the waterproof boxes that contain the electronics. Unfortunately, even with proper locking technology in place to prevent casual bad actors, it is extremely likely that if someone had enough motivation they could sabotage or destroy our product. This is because our indicator system is left unattended for extremely long periods of time (6 months to a year) and is in extremely rural remote areas that don't have much law enforcement presence.

Cybersecurity:

The primary cybersecurity concern that we have is signal jamming of our wireless communication. We are not concerned with an eavesdropping attack against our plaintext radio traffic as we end up displaying all of the information anyways, so an eavesdropper could just go look at our signage module and save themselves the trouble of eavesdropping on our traffic. We are concerned however with someone jamming our LoRa communications module which would prevent the water measurements from being properly transmitted to the roadside module. For our situation, we plan on relying on FCC regulations and monitoring to enforce policies against jamming as we don't have the budget to implement jamming resistant hardware. We believe that this solution will work because we have been unable to identify a benefit for an attacker to run a jamming attack, which comes with a costly criminal penalty, to run the risk of jamming our system.

Implementation Details:

Solar:

The solar system consists of two main components: a solar panel and a solar charge controller.

For our solar panel we opted for a 44"x20" 100W panel. We have two of these panels, one on the streamside module and one on the roadside. These panels are responsible for charging the battery and also powering the Arduino during peak sunlight hours. Our panels will be mounted facing South with a couple of options for panel angle depending on the maintenance plan of the entity who is deploying the indicator. The first option would be to change the angle of the solar panel by season, with the winter angle being approximately 67 degrees and the angle during the summer being approximately 15 degrees assuming that our indicator will be primarily deployed in Iowa. After consulting various resources online, we found that a very common calculation method for finding optimal charge angle during winter was (latitude * .9) + 29 and the optimal charge angle for summer is (latitude * .9) - 23.5. While the guidance we give is for Iowa, in the operation manual we give more guidance on panel mount angle to anyone who is operating our indicator. However, if operators do not want to be responsible for changing the panel angle throughout the year, they can average the two values calculated previously which yields a 41 degree tilt angle for Central Iowa.

Our solar panel is connected to a charge controller that has built in optimizations for our battery types, lead acid and Nickel Metal Hydride. This charge controller hooks up to the battery, load, and panel to charge the battery and power the load when the battery is either full or there is enough power to both charge the battery and power the load. The charge controller has built-in overcharge protection, short protection, temperature protection, and overcharge protection. The charge controller also has different charging cycles to help optimize the charge and prolong battery life. The electrical connections and charge controller for this whole system are stored inside the weatherproof boxes housing all of our connections and electronics.

Battery:

The battery section of the project uses two different systems. On the sensor side we used 8-1.2V NiMH batteries. On the signage side we used a single 12V lead acid battery.

For the first system using the eight NiMH batteries, we decided to use eight batteries because with them in series it grants us 9.6V. This was important because the Arduino that we are using operates correctly in a 7V-12V range. When it came down to which type of batteries we were going to use, we needed something with a wide temperature range as well as efficient rechargeability and a long lifespan. NiMH batteries have a very low temperature operating range which has high importance in places like Iowa. The batteries work perfectly down to -20 degrees celsius and operate down to -40 degrees celsius. Addressing the other important characteristic, these batteries have the ability to be recharged over 1200 times. They also keep 70% power even when they are not used for three years.

On the other side of the system with the 12V lead acid battery, we made that decision off of capacity. We based our design on signs that display driver speed in school zones, which have very similar power requirements.. That system requires a similar amount of power and is more accessible for battery changes. This battery will have less of a lifespan compared to the other side of the system, however, with it having a 22Ah capability we decided to use this big battery for the higher power intensive side. Both sides give our system more than enough voltage and current for everything to work optimally.

We also chose to use a simple voltage divider on the 12V lead-acid battery to make sure that we cut down some of the voltage and current entering the Arduino's as overloading them can cause heat problems and potential issues with the 3.3V and 5V output pins on the Arduino Mega board.

Wireless Communication:

The wireless communications side of the project consists of 2 main components: The transmitter and the receiver.

For our specific application we have decided to use Long Range (LoRa) technology since it supports long range coverage at a low power and transfer rate compared to other technologies currently out in the market. Some specifications to note that stood out in the REYAX RYLR896 transceiver module we chose has a maximum range of 15 km and a low operating temperature of -45°C. These two features are most important because if the system were to be placed at a stream and the nearest road is quite a distance away, our coverage range would be able to fulfill the needs. Also, since our system is designed for rural Iowa, the cold winter weather must be accounted for during this period. Some additional features that the components consist of are AES128 Data Encryption which we chose not to implement as explained in the security section, and excellent blocking immunities.

Being FCC certified, the module transmits at a band of 915MHz. Both the transmitter side and the receiver side operate at 3.3V, but the transmitter has an operating current of 43mA and receives at 16.5mA.

Utilizing all these features, we were able to transmit the water depth data from the ultrasonic water depth sensor to the receiver end to light up the signage system. This is done by using the AT commands for this module to send a string of data to the receiver which will parse out what value we need to light the signage.

Water Sensor:

The method we used to measure the water level was to use a DFRobot A02YYUW Waterproof Ultrasonic Sensor. The rationale behind this choice of a sensor is that unlike other methods of obtaining water level, this doesn't require the hardware to be in the water itself. Because of this there is less maintenance required and the sensor is more reliable.

The A02YYUW sensor has an integrated signal processing unit that allows the user to directly obtain distance measurements through an Asynchronous Serial Interface. With a baud rate of 9600 bit/s this allows almost any microcontroller to communicate with the sensor. Operating at a voltage of 3.3-5V and an average current of under 8mA this is an ideal choice for being powered by an Arduino.

Due to the outdoor nature of this project the sensor is IP67 waterproof and can operate in a temperature range of -15-60 ℃. We found that the sensor also has a fairly narrow beam pattern that also suits it well for our application.

Signage/LED:

The signage side consisted of many components. The components being the actual road sign and pole, the addressable LED strip, and an arduino mega 2560.

For the road sign we opted to use a 24” diamond sign from a road sign distributor. We chose a yellow diamond sign because according to the State of Iowa’s Department of Transportation, the standard for a road warning sign is a yellow diamond sign. With the sign we also had to consider durability and reflectiveness. Our sign is .08” Engineering Reflective Grade Aluminum, meeting ASTM D4956(Type 1) retro-reflective standards. Our sign’s outdoor durability is also good for up to 7 years, and can withstand temperatures in the range of -30 degrees Fahrenheit to 130 degrees Fahrenheit. This sign also has resistance to fading and mild chemicals. We also opted for the 24” sign versus the 18” to ensure higher visibility, however, we decided against using a 36” sign due to budget constraints. For the road sign pole we opted for an 8’ galvanized post coated in green enamel, in order to prevent rusting and corrosion.

For the addressable LED strip we chose to use WS2812B LEDs. The reason for this is these lights are very easy to program, have excellent brightness, and optimal battery consumption, and are also cost effective. We utilized the FastLed arduino library to program the LED’s. Our strip contains 60 individual addressable LEDs, each of which uses 60 mA of current, resulting in a total usage of 60 LEDS * 60 mA = 3.6 Amps of current. To make the strip resilient to weather conditions we are coating the LEDs in clear acrylic spray to waterproof the lights.

Finally the last component of the signage side is the Arduino Mega 2560. We opted to use this version of the arduino due to the fact that we need many pins to integrate the LED logic with the loRa receiver. Also, this Arduino offers the ability to draw power from a battery at a lower current than the Arduino Uno (Mega ~73 Amps whereas Uno is ~98 Amps for a 9V power supply).

Waterproof Boxes:

We opted to use two waterproof containers to hold our electronics when our prototype was implemented in the field. On the roadside module, we used an IP 54 rated box that is dust resistant and water resistant. As there is no risk of submersion here we didn’t need as tightly sealed of a box as we did on the streamside module. This box had water resistant ports that we were able to run wires from the solar panel and to the LED lights through. This box was also big enough to hold our battery.

For the streamside module, we used an IP 67 rated box, which means that it is dustproof and submersible up to 1 meter. This box also has ports that can be used to run wires through which we took advantage of for the sensor and solar panels.

Testing Process and Results:

Testing process:

Our testing process consisted of three main stages: unit testing, integration testing, and system testing.

Unit Testing:

Our team conducted unit testing on each of our main modules, for which we will go into more detail below:

- Ultrasonic Depth Sensor: The ultrasonic depth sensor was tested in still water with a solid bottom, still water with a substrate bottom (sand/rock on bottom), moving water with a solid bottom (agitated by hand), and in a stream which had moving water and a substrate bottom. For these tests the main data we were looking for was collecting a depth reading and comparing it to a manually measured depth which we captured by using a tape measure or yardstick. Our goal was to have the sensor be accurate to within 1.25 cm in all of our measurement tests.
- LoRa wireless communication module: We first tested this component connected to an Arduino in a lab to ensure that the LoRa radios were synced up and transmitting/receiving properly. After completing this step, we took the transmitting module to different places on campus and in Ames and sent data packets to the receiving module which was being monitored by another team member in the Coover lab. By traveling to different locations within a two mile radius of the lab, we were able to ensure there was a variety of different objects and competing signals mixed with our transmission. We sent 10 packets from each location and recorded whether or not they were received by the other LoRa module. Our desired receival rate was 90%.
- LED/Signage: We first used Department of Transportation resources to determine the size of the lettering on the sign. Then we verified these recommendations with our actual sign and made sure the visibility held up. For the testing of the LEDs we used a subjective testing method. We set up our programmed LED strip and first had our team members go to set distances away and rate how visible it was. We then asked our friends and classmates to let us know how far away it was for them to be able to see the LEDs. This helped let us know if the LED lights we were using were visible.

- Solar Panel: For testing the solar panel we wanted to be sure that we tested in as diverse of weather conditions as possible. This involved testing the solar panel in both sunny and cloudy conditions at different times in the day. We made sure to test in the early morning, afternoon, and early evening along with different weather conditions including sunny, partly cloudy, and cloudy. We then used these measurements along with weather charts showing the average number of sunny hours per day to ensure that our solar system would keep the batteries charged.
- Batteries: To test the batteries, we measured the current draw of our Arduino when it was fully connected and running and then conducted calculations to ensure that our batteries would be able to keep the whole system charged. We also plugged the components into our batteries and let them run for 24 hours to see the drop in battery levels.

Unit Testing Results:

The data collected during the testing phase and our interpretation of that collected data is shown below for each of our primary modules.

Ultrasonic water sensor:

Still Water Solid Bottom			Moving water solid bottom		
Sensor reading (cm)	Actual measured depth (cm)	Error Rate	Sensor reading (cm)	Actual measured depth (cm)	Error Rate
10.27	10.4	-1.25%	10.25	10.4	-1.44%
20.38	20.2	0.89%	20.15	20.2	-0.25%
25.28	25.3	-0.08%	25.37	25.3	0.28%
30.55	30.3	0.83%	30.39	30.3	0.30%
Still Water Substrate bottom			Stream		
Sensor reading (cm)	Actual measured depth (cm)	Error Rate	Sensor reading (cm)	Actual measured depth (cm)	
9.92	10.2	-2.75%	13.2	14.3	-7.69%
20.42	20.6	-0.87%	22.91	22.7	0.93%
25.03	24.7	1.34%	36.84	36.4	1.21%
30.36	30.4	-0.13%	43.1	42.2	2.13%

Our original goal for testing was to have the sensor be accurate to within 1.25cm (~1/2 inches) for all bottom types and water speeds. Our test data clearly demonstrates that our sensor is accurate enough to be sufficient in our design.

LoRa Communication Module:

In lab data:

In Lab/Coover		
Approximate Distance (m):	Packets received (out of 10)	Packets received
1	10	100.00%
5	10	100.00%
10	10	100.00%
20	10	100.00%
30	10	100.00%
50	10	100.00%

Interpretation: In a controlled indoor setting, our communication module is able to correctly receive all packets through walls and rooms containing electronic equipment that may be transmitting signals on a similar frequency.

In Ames/On campus			
Approximate distance (m):	Primary obstruction type	Packets received (out of 10)	
400	Buildings	10	100.00%
800	Trees	10	100.00%
1200	Buildings/trees	10	100.00%
1600	Buildings/trees	9	90.00%
2000	Buildings/trees	10	100.00%
2400	Buildings/trees	10	100.00%
2800	Buildings/trees	9	90.00%
3200	Buildings/trees	10	100.00%

Interpretation:

At various distances up to 3200 meters (~2 miles), our communication module is able to successfully send data packets through obstacles including buildings and trees. This data was collected on a cloudy cool day, so fluctuations in temperature, precipitation, and humidity could end up affecting the transmission success rate. However, according to numerous online resources and projects we examined, these fluctuations in meteorological patterns have a negligible effect on the effectiveness of the LoRa radio and protocol which is rated to transmit at up to 15 km.

LED/Signage

Test Data:

Subject #	Distance away LED is visible (approximate m)
1	45
2	35
3	60
4	50
5	75
6	25
7	55
8	60
9	40
10	35
11	50
12	55
Avg	48.75

Interpretation: Using NHTSA data, we are able to determine that at 55 mph, a driver needs about 150 ft (45 m) to slow down after recognizing an object. Our team made the assumption that a driver will already be slowing down if they plan to turn at the low water crossing, which increases the margin even further. Additionally, these readings were taken during lighter sunny conditions, meaning that at night or during storms the visible distance would be even higher.

Solar Panel:

Test Data:

Conditions	Wattage produced
Sunny (morning)	84 W
Sunny (afternoon)	91 W
Sunny (evening)	83 W
Partly cloudy (morning)	61 W
Partly cloudy (afternoon)	69 W
Partly cloudy (evening)	60 W
Cloudy (morning)	19 W
Cloudy (afternoon)	24 W
Cloudy (evening)	18 W

Interpretation:

Based on the weather charts we consulted, we can expect an average of 4.3 hours of peak sunlight per day in the darkest months of the year, December and January. Additionally, our solar charge controller will redirect additional power not being used to charge the battery to the load, which is our Arduino. Knowing these two things, we concluded that our solar system is strong enough to keep our battery charged and Arduino consistently powered.

Battery

Data:

	Wattage (W)	Voltage (V)	Current (mA)			
RF trans		3.3	43			
RF rec		3.3	16.5		Total Current Needed streamside:	Battery Life (hours, no recharge)
Signage		5	648	(adjusted since light will be flashing)	56	421.4285714
Sensor		3.3-5	8			
Arduino		7-12	5		Total Current Needed (mA) Roadside:	Battery Life (hours, no recharge):
					669.5	32.86034354

Interpretation:

We can determine from this data that our battery system will keep the system charged with little issue. Due to the low current draw of our system and solar capabilities that will both charge the battery and power the Arduino during peak light, we are confident that our system will be able to stay powered. We also recommend that the batteries be changed 1-2x per year in order to prevent reduced capacity due to recharging.

Integration Testing:

For integration testing, our team took a several step process to ensure that our different modules integrated seamlessly. For the first step, we integrated the solar panel and batteries. For the second step we integrated the water level sensor and LoRa transmitter.

Then, we integrated the LoRa receiver and LED system. Finally we connected these modules to our power system to complete assembly.

Battery and Solar:

For this step we connected the battery to the solar panel via our charge controller. We didn't collect much in the form of raw test data. Instead we verified that the data collected in the unit testing phase was still occurring when the units were integrated. We took time to put our solar/battery systems outside and measure the charge rate of the batteries during different stages of the day and weather conditions.

Water Level Sensor and LoRa Transmitter:

For this step we integrated the water level sensor and LoRa transmitter together on one of our Arduino boards. This involved taking the data received from the sensor and then transmitting via the LoRa sensor. Again, we ran under the assumption that because the data packets we were sending were nearly identical to our test packets, that the transmitter would still have the same relative range and integrity.

LED and LoRa Receiver:

For this step we integrated the addressable LED strip with the LoRa receiver. This involved programmatically ensuring that the packets from the LoRa were received and then sent correctly for display to the addressable LEDs.

Power System Connection:

In this system we connected the Arduino boards containing the other modules and connected them to our battery systems. We then ensured that enough power was being consistently delivered to all of the modules so that normal operation could consistently occur. We took measurements at the various connection points of the system to ensure that everything ran smoothly. We also placed the system outside and ensured that the charge controller was able to moderate all of the electrical components in the system properly.

System Testing:

This was the final portion of our testing for this project. After all of the components were assembled and waterproofed. We originally had arranged to test our system at Ledges State Park, but due to some concerns on their part we were unable to test there. We ended up testing the system in both a controlled environment and in brief periods outdoors.

Results:
To demonstrate the measurement capabilities of our system, we were able to test it on a bucket of water in a controlled setting and briefly on a small stream. To ensure that our power system was capable of providing the necessary power, we ended up charging the batteries fully and then connecting all systems operating normally and letting it run

without a charge. The signage module lasted between 28-32 hours on a single full battery charge while the detecting portion didn't end up fully depleting the battery during our testing window. Unfortunately, we connected our power system incorrectly during system testing which ended up burning out our addressable LEDs, and we were forced to take our demo video with a backup set of lights that wouldn't actually be in use on our project.

Data Explanation:

```
Depth: 897
Red
Depth: 856
Red
Depth: 137
Depth: 354
Yellow
```

Here you can see a readout from our LED unit showing what the current water depth is in millimeters. Additionally, you can see how the LED lights are lit with the corresponding color listed below it. These lights flash until a new reading is received by the unit and then our color conditions are reevaluated.

```
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
AT+SEND=1,2,17
```

For this sensor data, we can see that the average depth the water surface is away from the sensor is 17 mm. This data is then transmitted to the LoRa receiver and subtracted from 914, which is the distance the sensor is away from the streambed if you follow the mounting instructions in our manual

```
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 17 mm
distance: 853 mm
AT+SEND=1,2,58
```

This readout shows that the water surface is 58 mm from the sensor. This corresponds to the depth of 856mm and red light being on.

```
distance: 756 mm
distance: 740 mm
distance: 729 mm
distance: 729 mm
distance: 729 mm
distance: 729 mm
distance: 729 mm
distance: 552 mm
AT+SEND=1,3,777
```

This readout shows a distance of 777mm between the sensor and water surface. This corresponds to the depth of 137 and our light being deactivated, as this is a normal depth and safe for a car to drive through.

```
distance: 553 mm
distance: 557 mm
distance: 557 mm
distance: 557 mm
distance: 557 mm
distance: 557 mm
distance: 553 mm
AT+SEND=1,3,560
```

This final readout shows a distance of 560 mm between the depth sensor and water surface. This corresponds to the receiver having a depth of 354 mm and the light flashing yellow, meaning it is unsafe for sedans but safe for larger vehicles.

Context of Related Products and Literature:

Our product fills a unique gap in the landscape of related products and literature. We were unable to find any academic research on low water crossings and how safe they are for vehicles to traverse. In terms of measuring streams, we found two similar systems, but each has gaps that our project fills. The first tool that is currently used to measure streams is a flow rate meter. A lot of these meters are linked to websites where you can check in on the current flow rate, but the flow rate tells you nothing about the safety of driving through a low water crossing. The flow rate could be extremely slow, but if the stream is 24 inches deep the stream is unsafe to drive on. The other tool currently used is a depth sensor, however these tools have no way to tell drivers currently on the road the condition of the low water crossing and how safe it is to cross. Our product uniquely measures water levels and displays the safety information to drivers in a real time fashion.

Appendix I: Operation Manual

What you will need:

- 12V 22Ah Sealed Lead acid battery
- 8 AA NiMh batteries
- Appropriate wires
- 2 PWM charge controllers capable of charging lead acid and NiMh batteries
- Arduino Mega 2560
- Arduino Mega 2560/Uno
- Ultrasonic Water Level sensor
- WS2812B addressable LED strip with 60 LED lights
- IP rated boxes large enough to hold components
- 6ft and 8ft galvanized steel road sign posts
- 24" road sign with desired text
- 2x Reyax RYLR896 LoRa communications modules
- 2x small breadboard
- 2x 100W Solar panels

Step 1: Charge controller setup

- Take your 12V 22 Ah battery and put the positive and negative leads into the positive and negative "BAT" terminals on the charge controller
- The charge controller should power up and display information about your batteries like battery voltage and life
- Take the positive and negative leads from the solar panel and secure them in the positive and negative "PV" terminals on the charge controller
- Insert two of the small wires into the positive and negative "load" terminals which will then go into the Vin and ground ports on the Arduino Mega, but do not attach them yet
- Take your pack of 8 AA batteries and repeat steps 1-4 above for this pack

Step 2: Arduino Setup

- Download both the scan code and LED code from our team Github Repo: https://github.com/T-Rebischke/low-water-crossing
- Make sure you have 2 Arduino devices, with at least one of them being an Arduino Mega. For our project we used an Arduino Uno for the sensor module and an Arduino Mega for the LED lighting module
- Open your Arduino IDE and upload the code to both devices via the USB connection, they are now able to be run from the batteries

Step 3: Powering the Arduinos from the batteries

- Take the output of your charge controller for the 12V 22Ah battery and plug it into the Vin port and GND port on the Arduino Mega
- Take the output of your charge controller for the 8 AA batteries and plug it into the Vin port nad GND port on the other Mega/Uno
- You should see both of these boards power light come on and the boards power up

Step 4: Preparing the LoRa modules:

- On a small breadboard, insert the pins of the Reyax RYLR896 into the board. This breadboard will be used in tandem with the Arduino Mega that is running the LED code on it.
- Wire the 3.3V output from the Arduino to the Vdd pin on the chip, the GND from the Arduino to the GND on the chip, and the TXD pin on the chip to the RX0 port on the Arduino.
- On a different small breadboard, insert the pins of the other Reyax LoRa module into the board. This board will be used in tandem with the Mega/Uno that is running the scanning code on it.
- Wire the 3.3V output from the Arduino to the Vdd pin on the chip, the GND from the Arduino to the GND on the chip, and the RXD pin on the chip to the TX0 port on the Arduino.
- Both of the LoRa modules are now set up and transmitting and receiving to each other since you already have the code uploaded to the boards

Step 5: Connecting the Ultrasonic Water Level Sensor to the Arduino:

- Connect the red wire on the ultrasonic sensor to the 3.3V output from the board
- Connect the black wire on the ultrasonic sensor to the ground output from the board
- Connect the blue wire from the ultrasonic sensor to port 11 on the Arduino Uno or corresponding port on the Mega
- Connect the green wire from the ultrasonic sensor to port 10 on the Arduino Uno or corresponding port on the Mega
- Your ultrasonic water level sensor will now be taking readings and sending them to the Arduino

Step 6: Connecting the LEDs to the Arduino Mega

- Connect the red (power) wire on your addressable LED strip into the 5V output pin on the Arduino Mega. This Arduino is the other one and should not be plugged into the ultrasonic sensor.
- Connect the white wire into the GND port on the same Mega that you connected the red power wire into
- Finally, attach the green wire into port 7 on the mega. This port will send commands to the lights allowing them to change colors

Step 7: Test

- At this point you should have the basics of the system set up. You can test the water level sensor on different levels of solid surfaces and the lights should either be off, flashing yellow, or flashing red depending on how close or far away the "water" surface is

Step 8: Mounting waterproof boxes:

- Take the first waterproof box you have, which should be large enough to hold the 12V 22Ah battery and necessary electronics, and drill two holes in the back.
- Use bolts to mount this box to the 8ft galvanized steel pole. Make sure you leave enough space to mount the 24" sign above the box
- Attach the 24" road sign
- Take the other box and mount it to the pole similarly by drilling into the back and mounting to the pole. However, the bottom of this box must be 54" up from the exact bottom of the pole.
- Next, Use the provided gaskets that came with the box and attach one on the side, this will allow for the wires from the sensor to come in and the wires from the solar panel to also come into the box
- Attach the ultrasonic sensor the the bottom of the box with screws, making sure that the bottom of the sensor is flat against the bottom of the box

Step 9: Placing batteries/electronics

- Place the batteries and electronics in you waterproof boxes then close them, this will create a waterproof environment for your electronics to operate
- If there is any spaces that aren't airtight/watertight at this point, caulk them and allow the caulk to fully dry

Step 10: Place the road sign and measurement module

- Place your road sign in your desired location on the road near the turnoff for the road to the low water crossing. This location must be in direct/nearly direct sunlight for most or all of the day. Then bury the bottom of the sign 2 ft underground, leaving the top 6 ft exposed

- Place your measurement module 18" into the streambed on the edge of the crossing, with the ultrasonic sensor over the roadbed, also in direct sunlight

Step 11: Mount the solar panels

- On each post, mount the solar panels facing south at your desired angle. For central Iowa our group found that about 42 degrees is the optimal angle for year round solar charging, but there are many online resources that can tell you the optimal angle for the zip code in which the modules will be deployed

Step 12: Test and maintenance

- Ensure that the system is working as intended
- Change the batteries and clean the solar panels about every 6 months-1 year to ensure peak performance

Appendix II: Lessons Learned

- Underestimating timelines: Our team definitely underestimated the time that a lot of our steps would take which led to us scrambling near the deadline to get things finished. In the future we would make sure to better estimate timelines and hold ourselves more accountable.
- Interoperability: Our team had some issues with interoperability that slowed down our timeline and increased the project cost. In future projects we will make sure we do extensive research on the interoperability of modules in a project.
- Need for clear setup instructions: While system testing, we ended up incorrectly hooking up our power system, which led to the addressable LED strip burning out. This taught us about the need to have a clear list of steps needed to setup our project every time we tested it.

Appendix III: Code

Our project code can be found here: https://github.com/T-Rebischke/low-water-crossing

Appendix IV: Future Work

While our project completed a proof of concept that this technology is feasible, there is also a lot that can be improved upon for future iterations of the project. Below is a list of recommended improvements that we have for further development:

- Migration from Arduino to custom PCB. If staying with Arduino then move to a smaller board such as a Nano or Uno
- Reduce size of solar panels
- Custom make the waterproof boxes
- Better optimized LED light strips

- Add more strips for better LED visibility

Appendix V: Costs

Over the course of the project our team spent money on the following items.

Ultrasonic Sensor- $17.88

Arduino Mega- $48.4

Addressable LED Lights- $23

LoRa Radio Chips- $40

Arduino Mega- $48.5

12V 22Ah Sealed Lead Acid Battery- $65

8AA NiMh batteries- $18

2x 100W Solar Panels- $170

Battery Holder for AA batteries- $12

PWM Solar Charge Controllers- $32

Assorted cables- $45

Waterproof boxes- $55

Custom Road Sign- $77

8 ft Galvanized Steel road post- $47

6 ft Galvanized Steel road post- $40

Assorted hardware- $3