**Testing & Evidence**

Testing Methods

The goal is to determine how effectively the sensors can measure different water conditions and provide accurate real-time data.

We initially planned to use an Arduino MKR WiFi 1010 as our microcontroller. However, we quickly discovered that this board only supports 3.3V, whereas our sensors require 5V to function correctly. Due to this voltage incompatibility, we had to switch to the Arduino Uno R4 WiFi, which supports 5V and was expected to work better with our sensors.

Once we had the appropriate microcontroller, we started testing the turbidity sensor (SEN0189) while postponing the setup of the temperature sensor, as it required a resistor that we did not initially have.

For our initial tests, we selected two water samples with significantly different clarity levels:

Tap water – expected to be clean with minimal turbidity.

Coffee – a high-turbidity liquid used to simulate dirty or contaminated water.

These contrasting samples allowed us to evaluate how well the sensor could differentiate between clear and turbid water.

Turbidity Sensor Testing

To begin, we researched online tutorials and documentation for the SEN0189 turbidity sensor. Most sources suggested connecting it to an analog input on the Arduino. However, when we followed this method, our readings were unexpectedly binary (only 1s and 0s) rather than providing a range of values.

Suspecting a connection issue, we tried switching the sensor to a digital input, which led to slightly better results (0 – 1023). However, the values still did not vary as expected between different water samples. The value stayed the same despite we tried it in different substances. This prompted further troubleshooting, where we investigated how the sensor calculates turbidity values.

We suspected that the SEN0189 turbidity sensor was faulty. To verify this, we ordered a different turbidity sensor, the Jopto TSW-30 Turbidity Sensor, expecting better results. Unfortunately, this new sensor also failed to provide valid readings under different conditions.

After further investigation, we assumed that these turbidity sensors are not compatible with the Arduino Uno R4, so that’s why we decided to switch this sensor to Arduino Yun which we had in our lecture’s cabinet. Unfortunately, this didn’t give us results either.

Temperature Sensor Testing

Once we received the required resistor, we attempted to set up the DS18B20 waterproof temperature sensor. We used multiple wiring configurations and different code variations from online sources.

Despite our efforts, the sensor failed to function correctly. The main errors encountered were:

* "Sensor not detected" – indicating a connection or code issue.
* A constant reading of -127°C – which typically occurs when the sensor is not being detected at all.

At one point, a wiring mistake caused the sensor to become extremely hot, likely due to the wires being placed too close together on the breadboard. This highlighted the importance of precise wiring when working with temperature sensor.

Since the DS18B20 requires a 4.7kΩ pull-upresistor between VCC and the data pin, any deviation from this setup can result in errors. Additionally, ensuring the correct OneWire and DallasTemperature libraries are installed is crucial for the sensor to function.

Combination of Parts Together

During this phase of the project, we used a completely new setup. We created a grounded circuit on a breadboard to collect data from both the turbidity and temperature sensors.

On our first attempt, incorrect wiring caused the resistor to become extremely hot. After troubleshooting, we identified the issue, repositioned the resistor, and replaced it with a new one. Once the wiring was corrected, we successfully obtained data from both sensors and even managed to send and receive it using the Blynk system.

A row of resistors on a grey surface

AI-generated content may be incorrect.

Results

As the successful part of our project, we integrated the system with Blynk (an IoT platform) to enable real-time monitoring and remote access to water quality data. This will allow us to display turbidity levels dynamically, improving usability. The sensors were connected but the calibration failed and they do not match the required values.

Challenges met during the testing

Throughout the testing phase, we encountered several challenges and areas for improvement:

Microcontroller Compatibility Issues: The Arduino MKR WiFi 1010 only supports 3.3V, making it incompatible with our sensors, which require 5V. The Arduino Uno R4 WiFi, while resolving the voltage issue, turned out to be incompatible with our turbidity sensors, causing ongoing sensor malfunctions.

Turbidity Sensor Difficulties: Initially, the sensor output was just 1s and 0s in analog mode. Switching to digital mode improved results but still lacked proper variation. Introducing a resistor to fix the 0V issue led to fluctuating readings (1-10 and 500-1000), which were incorrect. The Jopto TSW-30 Turbidity Sensor was also tested but did not work correctly.

Temperature Sensor Setup Failure: Despite trying multiple configurations, the DS18B20 sensor was either not detected or displayed -127°C. One incorrect setup caused the sensor to become extremely hot due to wiring mistakes on the breadboard. The sensor requires a 4.7kΩ pull-up resistor to function correctly.

Multiple sources and wiring setups were tested, but none provided a working solution for the temperature sensor.

**Future Steps**

To improve the accuracy and reliability of our water quality monitoring system, we will attempt to connect a fully functional turbidity sensor. Given our previous challenges with sensor compatibility, we will carefully test different configurations and ensure proper calibration. Additionally, we will test a wider range of water samples beyond just tap water and coffee, including filtered water, rainwater, and potentially water from natural sources such as rivers or lakes. This will allow us to assess the effectiveness of our system under varying conditions.

For the temperature measurement component, we will double-check the DS18B20 sensor wiring and verify that the correct pull-up resistor is used. Since incorrect wiring previously led to sensor malfunctions and overheating, ensuring proper setup is crucial. If necessary, we will explore alternative temperature sensors that may be more compatible with our system.

To enhance the project's functionality, we will continue refining data logging and visualization. One potential improvement is integrating the system with Blynk or a similar IoT platform, allowing for remote monitoring of water quality parameters. This would make our project more practical for real-world applications, enabling users to receive real-time updates on water conditions.

Furthermore, we aim to expand our project by integrating a pH sensor, providing additional insight into water quality. Measuring pH levels alongside turbidity and temperature will give a more comprehensive assessment, making our system more valuable for applications such as environmental monitoring and drinking water safety.

We also plan to enhance the physical design of our project to improve stability and portability. Soldering the wires will ensure more stable connections, reducing the chances of loose or unreliable contacts. Additionally, gluing or securing all the components together in a compact structure will make the device easier to handle and transport. These improvements will contribute to a more durable and professional-looking final product.

Incorporating AI is another step we plan to take. By utilizing machine learning, we can analyze sensor data and improve the accuracy of our assessments based on user needs and environmental conditions. AI-driven algorithms could help detect patterns, predict potential water quality issues, and provide recommendations for corrective actions.

Overall, this project highlights the importance of hardware compatibility, calibration, and troubleshooting in sensor-based systems. By refining our methods, improving the physical design, and expanding our testing conditions, we aim to develop a reliable and practical water quality monitoring tool. Our goal is to create a system that can be used in real-world scenarios, contributing to better water safety and environmental awareness.

This project demonstrates the importance of hardware compatibility, calibration, and troubleshooting in sensor-based systems. By refining our methods and expanding testing conditions, we aim to create a reliable water quality monitoring tool that can be used for real-world applications.