THE UNIVERSITY OF BRITISH COLUMBIA

IoT Monitoring of Aquaponic and Hydroponic Food Production

Verification and Validation

UBC Electrical and Computer Engineering Capstone 121

Carson Berry
Lynes Chan
Mason Duan
Jayden Leong
Hannah Xu

for

Nelly Leo, UBC SEEDS Sustainability Program
Bernhard Nimmervoll, UBC Mechanical Engineering

February 21, 2021

Contents

1 Change Log

Date	Author	Sections	Change
2020-10-05	JL	1.0 - 4.0	Creation of document skeleton
2020-10-16	MD	1.0 - 4.0	First draft
2020-11-28	CB	2.0 - 4.0	Added subsystem specifics test outlines
2021-01-15	CB	3.0	Restructuring document to reflect design and requirements
2021-02-19	CL	1.0 - 6.0	Added section 6.0 Validation and minor changes to earlier sections to be consistent with the requirements documentation
2021-02-19	JL	2.0 - 3.0	Reworded overview section to be more concise. Updated testing section to be more concise, and match updated functional requirements. Removed some outdated tests due to changed requirements.

2 Overview

This document outlines the verification and validation plans for our project, an IoT hydroponic or aquaponic monitoring system. The verification tests and results included in this document seek to prove that our project meets its requirements. For more information about these requirements, please see the included *Requirements* document.

The design of our project informs the structure of the verification testing and results below. Our hydroponic and aquaponic monitoring system contains four main components: the software, the internet of things (IOT) infrastructure, the sensors, and the controls. As such, our verification tests and results will be organized under each of these subsystems.

Additionally, this document outlines our project's validation plan, to ensure the project meets our client and user's needs. To see our validation tests and results, please see the concluding section of this document.

3 Verification

This section contains detailed subsystem tests verifying that functional requirements are met. Table 1 below indicates the corresponding verification tests for each of the functional requirements. Note that each of the verification tests is marked with a abbreviated label according to its subsystem. The following list gives of the meaning of each of these abbreviations:

- ST_x = Software test x
- IT_x = IOT test x
- SET_x = Sensor test x
- HW_x = Hardware test x
- CT_x = Controls test x

Table 1: Functional Requirements Verification Test Overview.

Functional Requirement	Verification Tests
$F1$	ST_2, ST_4 IT_1, IT_2, IT_3, IT_4, IT_5 SET_1, SET_2, SET_3, SET_6
F_2	ST_1, ST_3 IT_2, IT_3
F_3	IT_2, IT_4 HT_1, HT_2
F_4	IT_1, IT_2, IT_3, IT_4, IT_5 CT_1, CT_2, CT_3, CT_4 HT_1
F_5	IT_4, IT_6 HT_1
F_6	SET_1, SET_2, SET_3, SET_4

In the following subsections, we will describe the tests from Table 1 in detail. Additionally, results of these tests will be stated.

3.1 Software Testing

The following sections describe each of the software tests, and their corresponding results.

$\mathbf{ST_1}$: Raspberry Pi communicates data without loss to the user application

The raspberry pi's connection to the internet should be verified upon installation. This is accomplished by powering on the device and SSH-ing into the device. Once that is done, the *ifconfig wlan0* bash command should be used to test what wifi connection, if any, is in use. The webpage https://www.raspberrypi.org/documentation /configuration/wireless/wireless-cli.md can be helpful in troubleshooting wifi connection if necessary. Once this is done, then you can run the *verify_connection.py* program using python.

Results: After installation, the pi was able to run *verify_connection.py*

ST$_2$: Displaying Most Recent Data Polled From Sensors

While the software application is running, the software application should display the most recent data polled from the aquaponic sensor. This is tested incrementally, per sensor, by opening the application and making sure that way the data is displayed on the screen matches the layout that is displayed in our software design mockups. We test the data validity by comparing the data displayed on the software application to the one that is stored in the IOT system.

ST$_3$: Displaying Historic Data Gathered From Sensors

The user should be able to choose to see the historic aquaponics data gathered from the aquaponic sensors when chosen to do so. We test this instrumentally by manually requesting to display the historic aquaponics data gathered from the aquaponic sensor (via some kind of input command) and making sure that way the data is displayed on the screen matches the layout that is displayed in our software design mockups. The data validity is known by comparing the data displayed on the software application to the one that is stored in the IOT system.

ST$_4$: Sending Sensor Reading Request to IOT System

The software application should be able to send a signal to the IOT system to initiate a manual read of the aquaponic sensors. This is tested by manually requesting info from the sensors and making sure that the software application is able to receive and store this data. During installation, this will be verified by using calibration sensor environments. For more information see below on sensor calibration.

The data validity is tested by checking the IOT system and ensuring that a signal to trigger the sensor reading has been received.

ST_5: Sending Control Request to IOT System

The software application should also be able to send a signal to the IOT system to initiate manual activation of the aquaponic controls. This is tested by requesting to send a control request to the IOT system through the application. This test should be conducted using a device that is known to be working, and requires 120V AC power. This could be a lamp, or a cell-phone charger, or the water pump that is part of this project. This test is designed to test if the devices on the power bar can be controlled from the application. Send an 'on' request from the application, and if the device does not turn on within 1 minute, check that your hub is connected to the internet.

3.2 IOT Testing

The following sections describe each of the IOT tests, and their corresponding results.

IT_1: Handling Requests from Software and relaying command to Sensors and Controls

When the IOT system receives a request from the software application to read the sensors, the IOT system should trigger the activation of the aquaponic sensors for a read of the current aquaponic data. We test this instrumentally by manually running the function that would normally run when the IOT system receives a request to read from the sensors. We check that the sensors are then activated and return a read value back to the IOT system. We test the validity of the data received by the IOT system from the sensors by comparing it to external measurements of the aquaponic data.

IT_2: Receiving and Storing Data Updates from Sensors

The IOT system should be able to receive data read from the sensors, and then store the data. We test this instrumentally by manually running the function that would normally run when the IOT system request for a data read from the sensors.

We check that the sensors are then activated and returns a read value back to the IOT system, which would then be stored. We test the validity of the data stored by the IOT system from the sensors by comparing it to external measurements of the aquaponic data.

$\mathbf{IT_3}$: Sending Requests to Sensors to Update Aquaponics Data

The IOT system should be able to send periodic requests to the sensors for a read and should be able to update the data stored with the new data received. We test this instrumentally by manually running the function that periodically sends signals to the sensors. We check to ensure that it is activating the sensors are the desired intervals. We test the validity of the data stored by the IOT system from the sensors by comparing it to external measurements of the aquaponic data.

$\mathbf{IT_4}$: Always Operational

The IOT system should be operational and running 24/7. We test this instrumentally by plugging in the power to the IOT system and leave it running. We validate the test by checking the device periodically to ensure that the IOT system is still running and has not shut down.

$\mathbf{IT_5}$: Readily Available to be Queried

The IOT system should be able to be queried from a software application whenever. We test this instrumentally by sending dummy read requests using software application such as Postman. We validate the test by checking the IOT system and ensuring that the queried has been well received by the IOT system.

$\mathbf{IT_6}$: Able to push alerts to the software application asynchronously

The IOT system is able to push data onto the cloud servers. This is tested by the *push_data_to_cloud.py* script.

3.3 Hardware testing

The following sections describe each of the hardware tests, and their corresponding results.

HT_1: 12V and 5V power rails are operational

Using a digital multimeter set to DC mode, check that the 5V output of the linear regulator is within 10 mV of 5V. The 12V rail should also be within 50mV of 12 V.

Results: The linear regulator connected to the 5V power rail is running at 10mV and and the 12V rail is within 50mV.

HT_2: Run from batteries and boost circuit

The sensor hub must be able to run with battery power in the event of a power outage. To make this design user friendly, as documented in the design document, our sensor hub will have backup batteries. The battery system will include a boost converter and specifications for a charging circuit that will not be tested due to safety limitations associated with the use of LiPo batteries. This circuit should be able to produce 5V power to the MCU and sensors for at least an hour. The lifespan of this circuit in normal operation will be recorded for the design data sheet.

Results: Testing the backup power system is accomplished in 3 stages. First, the 5V boost converter board is connected to 2AA batteries to verify that the boost converter is able to output 5V DC. This can be checked with a multimeter, by connecting the positive probe to Vout, and the negative probe to GND. Second, the 5V boost converter shutdown circuit's functionality is also tested. The circuit is first modelled in a circuit simulator program to verify it has the desired functionality of outputting a low voltage when the 12V power supply is present and a 'high' voltage (above 0.8V) when the 12V power supply is absent. A plot of this behaviour can be seen in figure 2. Figure 1 is also provided for reference.

Once this behaviour is verified in a circuit simulator, then the circuit should be constructed. To verify the circuit's functionality, a multimeter voltage probe is placed at the SHDN node (collector of the BJT) and GND. When the 12V power

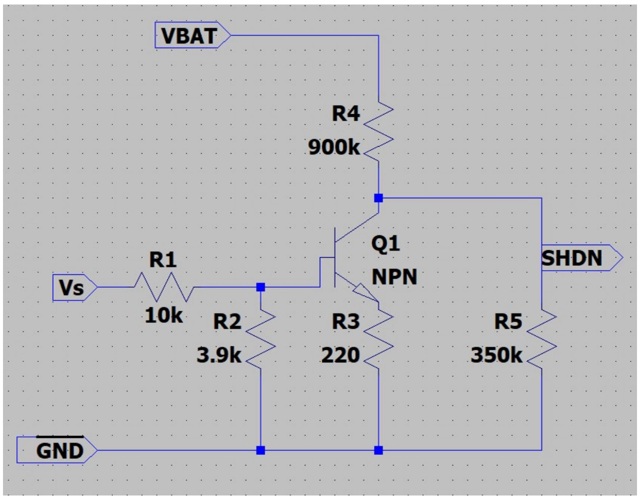

Figure 1: Boost converter shutdown circuit. This circuit disables the AA batteries when 12V is present.

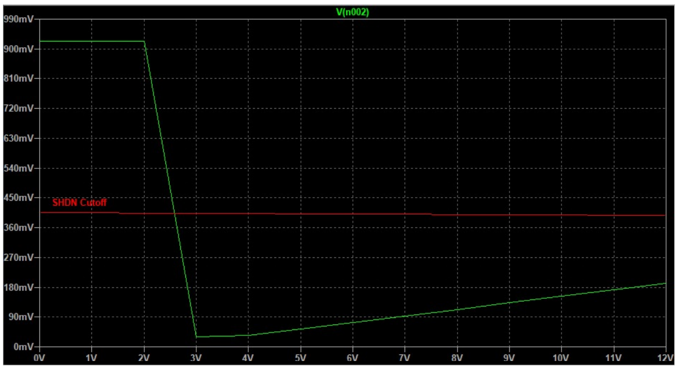

Figure 2: 5V boost converter shutdown circuit output vs source voltage. We see that once the source voltage (normally 12V) drops below 3 V, the SHDN signal goes high.

source is present, 100mV should be be seen. When the 12V Vs power supply is disconnected, 800mV should be observed.

Finally, the overall backup power system can be tested once it is constructed according to the design document. This is done by attaching a multimeter voltage probe to the 5V rail and GND, and plugging in the 12 V power source as well as connecting 2AA batteries to the 5V boost converter. During normal operation, that is, when 12V is present, the voltage rail should read 4.7V. When the 12V power supply is disconnected, or unplugged, the voltage rail should now read 5V. This difference is because the 12V power supply is connected through a schottky diode after it is converted to 5V. Additionally, to ensure the battery is not being used when the main power supply is operational, an ammeter can be added in series between the battery and the 'Vin' pin of the 5V boost converter.

3.4 Sensor Testing

The following sections describe each of the software tests, and their corresponding results.

$\mathbf{SET_1}$: Measuring the Different Data in Aquaponics System

The sensors should be able the measure the different data that exists within the aquaponic system: temperature, alkalinity. We test this instrumentally by deploying the sensors within different calibrated environments and activate the sensors to read and ensure that we get a reading back. We test the validity of the data by comparing the measured readings to what we know the value to be in our calibrated environments.

$\mathbf{SET_2}$: Measure temperature within 1 degree of correct temperature, with 0.1 degrees of precision

The temperature sensor should be placed in a bucket of ice-water, the temperature of which can be known to be 0 degrees. Calibration changes can be made on the mobile application by navigating to the "sensor-calibration" page and selecting "temperature".

$\mathbf{SET_3}$: Measure pH within 0.5 pH of correct pH, with 0.1 precision

Place the pH sensor in a cup with distilled water in it. The pH of the distilled water is known to be 7, with 0 electro-conductivity. Calibration changes can be made on the mobile application by navigating to the "sensor-calibration" page and selecting "pH".

SET_4: Correctly identify low water levels at least 95% of the time

Attach the sensor using the attached adhesive pads in the desired location that water should never be lower than on the tank. This location is typically in the top 15% of the tank. When there is no water present, press the "set" button on the breakout board. This is done to recalibrate the baseline capacitance. Place a plastic cup against the inside of the tank above the level of the water, forming a seal with the wall of the tank. Slide the cup down into the water to the level of the water sensor, moving it over the portion of the tank that the water-level sensor is attached to (but on the other side of the tank). Verify that the low water level alert is activated, and that the pump turns on. If the low-water level alert is not activated, verify that the cup is indeed full of air, not water, and is located on the other side of the tank from the sensor. If the sensor is still not correctly registering the low-water level, try actually lowering the water level of the entire tank by draining some of the water. If this does not work, the sensor should be removed from the tank exterior and the re-attached to a clean location on the exterior. The original cup-test should be repeated.

SET_5: Correctly identify leaks from the aquatic system at least 95% of the time

The leak sensor's function can be verified by placing it on a flat surface and, pouring a cup of tap-water near the sensor, so that some of the water gets the leak sensor wet. You should be notified by alert, that there is a water leak at sensor 1.

SET_6: Reading From Sensors When Receiving Read Request

The sensors should be able to a react to read request from the IOT infrastructure. We test this instrumentally by manually sending a read request to the sensor and ensure that the sensor reacts by reading the aquaponic data. We test the validity of the data by comparing the measured readings to what we know the value to be in our calibrated environments.

$\mathbf{SET_7}$: Waterproof

The waterproof sensor enclosure must comply with the rating of IP65. This means that the enclosure must be protected from low pressure water jets in any direction.

Results: After spraying a shower head to the enclosure for 15 minutes in all directions, no liquid was able to enter the enclosure.

3.5 Control Testing

The following sections describe each of the control tests, and their corresponding results.

$\mathbf{CT_1}$: pH controlling peristaltic pump is controllable

The peristaltic pump can be verified to be operating normally after installation by removing the outlet hose from the tank and placing the end of it in a cup. Toggling the peristaltic pump on from the application, the pump should begin to pump liquid into the cup within 60 seconds. If it does not, connect the pump up directly to the 12 V rail to verify it's function. If it does not work with 12 V DC, the pump is faulty and a new one should be obtained.

Results: The peristaltic pump starts pumping almost immediately after the on signal is sent. The pump is able to transport water from one cup to another.

$\mathbf{CT_2}$: Devices are able to be toggled remotely in 1s intervals

The test-signal routine on the mobile application will verify that the system can toggle on and off in 1 second intervals.

Results: Devices respond within 1s intervals after a test-signal is sent.

CT_3: Able to use actuators (pH peristaltic pump, water valve) to control the different aspects of the aquaponics system

Due to the inertia of the large volume of water present in any aquaponic/hydroponic system, it will take considerable time/input to affect the parameters of the aquaponics system with the control devices. The function of each actuator is verified in it's ability to affect the overall system within one hour by running the "self-test" function from the mobile application. This function cycles through each of the sensors, to verify that each is functioning and the system is responding to them within one hour.

4 Validation

This section contains the validation tests for the client. The main purpose of these tests is to make sure the user is able to navigate through our system with minimal difficulty. These tests are designed for users who have read the user manual. The user manual must be read before beginning these tests. For design details, please refer to the Design Key Document. For system-level requirements, please refer to the Requirements Key Document.

V_1: Viewing the status and the values for pH and temperature of the system

The user must intuitively navigate (with little to no instruction) through the app to find the water level and the leak statuses. The user must also be able to find the current value of the pH and the temperature through the app. The user successfully passes this validation test if they're able to achieve what is mentioned above with little to no instruction.

V_2: Customizing the values of the desired pH of the system

The user must intuitively navigate through the app to changed the desired pH of the system. If the user achieves this with little to no instruction, then the user successfully passes this validation test.

V_3: Access previous data

The user must intuitively navigate through the app to find past data that has been stored in the cloud. The user must understand the data in the form of graphs and tables and be able to see a trend if there exists one. If the user achieves this with little to no instruction or clarification on the data shown, then the user successfully passes this validation test

V_4: Calibrate the sensors and actuators

After reading the user manual, the user must be able to calibrate the sensors either through a physical knob (pH) or within the code (temperature). The user must be able to calibrate the actuators through the open source software provided. If the user achieves what is mentioned above, then the user successfully passes this validation test.