LAGRANGIAN WATER SAMPLER

PROJECT DELIVERY – FINAL DELIVERY

PREPARED FOR:

CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

Prepared by:

Richard E Scafidi – Software Engineer

richard@scafidi.dev

TABLE OF CONTENTS

ORIGINAL PROPOSAL

EXECUTIVE SUMMARY

The Lagrangian Water Sampler (LWS) is a custom-made device, created under the direction of California State University, Northridge (CSUN), who is hereon known as the Customer in this proposal. The LWS is used for automated collection of water samples at specific time intervals throughout each day, indefinitely while the onboard battery is still powering the device. The device was designed to be fully submerged under water at depths of approximately two to three meters. The device uses an Arduino based microcontroller and electronic components to control two water pumps on specifically programmed intervals. The device is powered by several large removable batteries. Recently, the device has become inoperable and troubleshooting attempts that were made to restore the device were not successful. CSUN is requesting to have the device restored to original functionality, and to provide technical documentation for the electronic circuitry, component design, and microcontroller code maintenance procedures to facilitate future maintenance efforts.

PROJECT SCOPE

The scope of this project is as follows:

- Examine, troubleshoot, and understand the original electronic circuitry.
- Design, assemble and test fixes to restore the system to functionality described in the Functional Requirements section.
- Document electronic circuit design and functionality.
- Document microcontroller source code and code flashing/building procedures.
- Document all electronic components within the system including part type, number and function/use case where possible.
- Testing is limited to electronic components and microcontroller programming only. The original waterproof housing will not be provided for testing and as such will not be a part of the test cases.
- There will be no continued support for the device after delivery of the functional device and documentation under this proposal.

PROJECT DELIVERABLES TIMELINE

The estimated timeline for deliverables is as follows:

Deliverable Name	Description	No Later Than Date
Feasibility Report	This is a single document that outlines the technical work that will need to be done after examining the device, and will deliver a statement on whether or not the planned project objectives can be delivered by the desired final deliverable date.	31 August 2022
Project Update Briefs	If the project continues beyond the Feasibility date, project update briefs will be held on a recurring basis. These will be written reports that outline the current state of the project including progress, delays, and any other information. It will also include demonstrations of any functionality in order to ensure the device is being designed to meet the customer's expectations.	Bi-weekly for the duration of the project
Post-testing Report and Demonstration	When the engineering team determines the project has met all documented requirements, a final testing report will be delivered. This report will be accompanied by a product demonstration to validate customer requirements have been met.	15 November 2022
Documentation Validation	If the device is validated by the customer, the next deliverable will be product documentation to include parts lists, electronic diagram documentation, troubleshooting documentation, and code documentation.	1 December 2022
Delivery of hardware and documentation artifacts	After the documentation is validated, the device will be returned to the customer along with all documentation artifacts. At this time the project will be considered closed.	15 December 2022

Changes to the project may affect the above timeline. Adding new requirements may push the project beyond the desired completion timeline of December 2022. A best effort will be made to inform the customer of any such potential impacts.

BUDGET

This project is estimated to require between thirty (30) to forty (40) hours of engineering time for troubleshooting, repair, documentation and testing of the product. It will also potentially require acquisition of new electronic components. The budget estimates are outlined below. For all material costs, an estimate is given now. At the time of delivery and payment, actual receipts for any material item will be provided and the cost will be adjusted to match. The material estimates below are conservative estimates.

Item	Cost per Unit	Quantity / Type	Total
Engineering Time	$45	35 Hours	$1575
Electronic components	$150	-	$150
Grant Total Estimated			$1725

If, after the project feasibility report, it is determined that the time estimated will vary plus or minus 10%, the Customer will be notified for approval to continue.

FUNCTIONAL REQUIREMENTS

Functional requirements are specific requirements that the system must meet to be considered complete. Below is a list of functional requirements.

- The system must be functional in an underwater environment at depths of two to three meters. This requirement will not be tested due to original housing unavailable for testing. No modifications are planned that would interfere with the original housing waterproofing capabilities.
- The system must run on the provided battery supply which outputs 12.5 volts.
- The system must use local time to operate on the following schedule:
    - Activation of Pump 1 from local time 10:00 AM until 4:00 PM
    - Activation of Pump 2 from local time 10:00 PM until 4:00 AM

NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are requirements not specifically related to a function of the device itself. Below is a list of non-functional requirements.

- The documentation provided will be in English.
- The documentation provided will include technical and non-technical aspects of the system, including each component, the function and use case within the system, and technical documentation for any source code provided.

DELIVERY REVIEW

DELIVERABLE SCHEDULE RESULTS

Deliverable Name	No Later Than Date	Outcome
Feasibility Report	31 August 2022	No report was generated. Verbal discussion with the customer resulted in simply fixing the functionality rather than redesigning the entire product. Due to the dramatic reduction in scope, a full report on how feasible a new product would be and what it would look like was not necessary.
Project Update Briefs	Bi-weekly for the duration of the project	We did not adhere to the bi-weekly updates on a regular basis, but we did provide casual updates on progress.
Post-testing Report and Demonstration	15 November 2022	This timeline was not achieved with the Customer, but extensive testing was done in-house by this date.
Documentation Validation	1 December 2022	All documentation was submitted to Customer on 28 November for validation.
Delivery of hardware and documentation artifacts	15 December 2022	One control unit will be delivered by 1 December and the second unit will be delivered by 20 December, pending any feedback from the Customer. Customer notified of final delivery and the new date is acceptable.

TOTAL COST TO DATE

Item	Actual Cost	Actual Quantity	Actual Total
Engineering Time	$45	32	$1,440
Electronic components	$190.37	-	$190.37
Shipping	$24.27	-	$24.27
Grand Total			$1,654.64

*Note that extra materials were ordered and will be delivered to Customer for future repairs if necessary. Some components ordered will be retained for testing.

USER ACCEPTANCE TESTING

One Lagrangian Water Sampler unit was returned to the Customer by 1 December to validate project deliverables. The project deliverables consist of all documentation, along with one unit to test in the field.

Field testing was conducted, and no issues were reported. User acceptance testing considered to be completed. Second unit shipped on 16 December and will arrive no later than 20 December.

PRODUCT DOCUMENTATION

Below is a summary of the documentation provided as part of this project.

Appendix A – Circuit Diagram

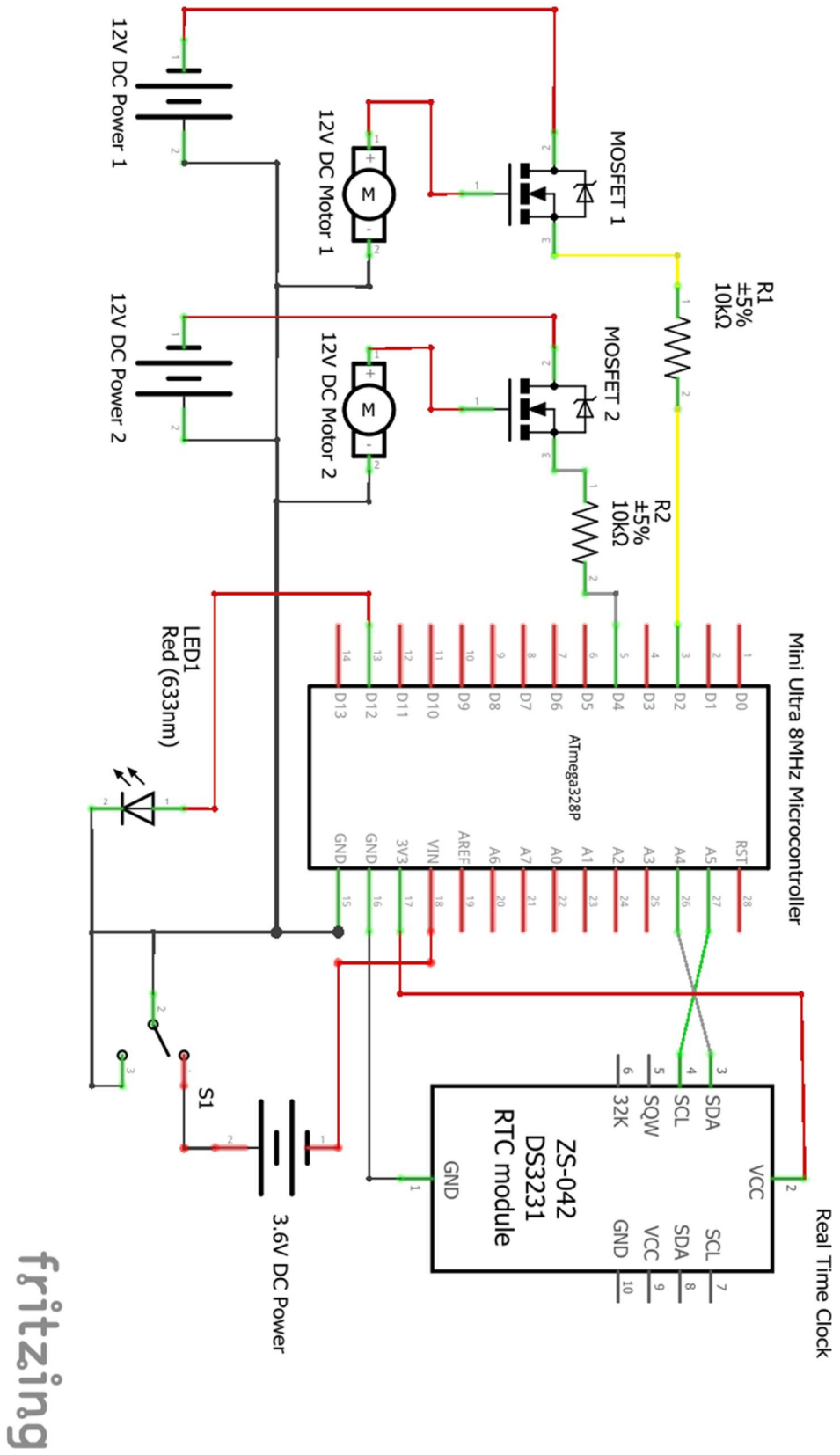
Appendix B – Testing Report

Appendix C – Troubleshooting Manual

Appendix G – Bill of Materials

Appendix H – Material Cost Receipts

APPENDIX A – CIRCUIT DIAGRAM



APPENDIX B – TESTING REPORT

CONTROL UNIT 1 TEST RESULTS

Component	Results
LED	Pass
MOSFET Transistor 1	Pass
MOSFET Transistor 2	Pass
DS3231 Real Time clock	Pass
3v Clock battery	Fail
Microcontroller pin 3	Fail
Microcontroller pin 5	Pass
Microcontroller VIN	Pass
Microcontroller FTDI Header	Pass
Microcontroller Ground	Pass
Microcontroller pin A4	Pass
Microcontroller pin A5	Pass
Microcontroller 3.3v Power Supply	Pass

CONTROL UNIT 2 TEST RESULTS

Component	Results
LED	Pass
MOSFET Transistor 1	Pass
MOSFET Transistor 2	Pass
DS3231 Real Time clock	Pass
3v Clock battery	Fail
Microcontroller pin 3	Pass
Microcontroller pin 5	Pass
Microcontroller VIN	Pass
Microcontroller FTDI Header	Pass
Microcontroller Ground	Pass
Microcontroller pin A4	Pass
Microcontroller pin A5	Pass
Microcontroller 3.3v Power Supply	Pass

The components were tested on both pump control devices provided. For one control device, the only defective part was the 2032 3v watch battery powering the DS3231 real time clock. For the other, there is a voltage leak within the A328P processor causing pin 3 to be in HIGH state while the board is powered. This in turn keeps the MOSFET open, which will keep pump 1 on. A replacement microcontroller was ordered and soldered in place to restore functionality to the second controller. The 2032 3v watch battery was also bad on the second controller. Both watch batteries were replaced.

The code was completely rewritten for several reasons. First, the real time clock can track when a power failure has occurred, a feature which was previously not utilized. This feature was incorporated into the refactoring of the code to put the controller into a state such that normal operation will not proceed until the problem is addressed. The LED on pin 12 will flash continuously, which should alert the user that something is wrong with the controller. The maintenance manual walks through how to troubleshoot and restore the device to working order.

The second reason for refactoring is to make the code more readable. Detailed comments have been provided within the code to explain key portions of functionality. Additionally, the time control structure was rewritten to make it simpler to understand and modify. Outside of the comments, the code itself has been written in a manner to increase self-documentation, which will make it easier for users to read and change if necessary.

The final change relates to the inclusion of preprocessor directives to change the program execution behavior by simply commenting or uncommenting single variables. This again makes it easier to put the microcontroller into various testing states as well as enhancing the method of syncing the clock. The following page contains steps used to test each component with the new code.

APPENDIX C – TROUBLESHOOTING MANUAL

TESTING DEPENDENCIES

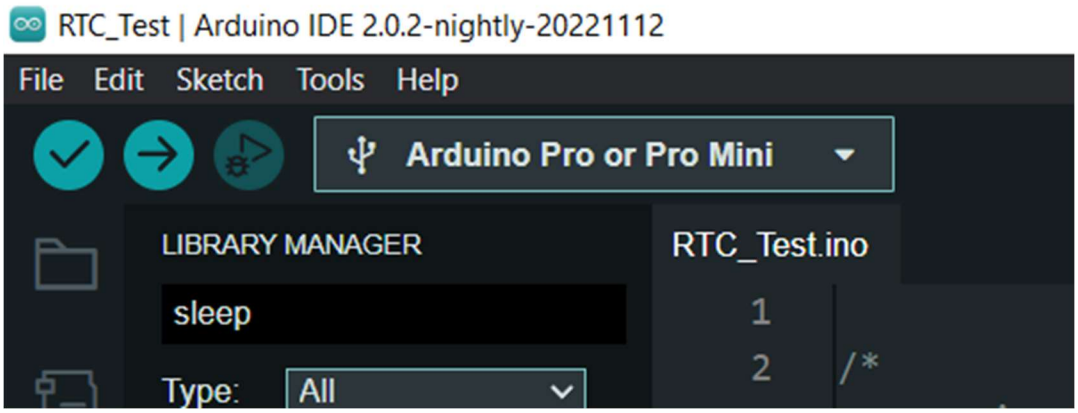
Dependencies used in testing:
Adafruit RTClib library v2.1.1
Adafruit SleepyDog library v1.6.3
Standard Wire library

FLASHING CODE TO THE MICROCONTROLLER

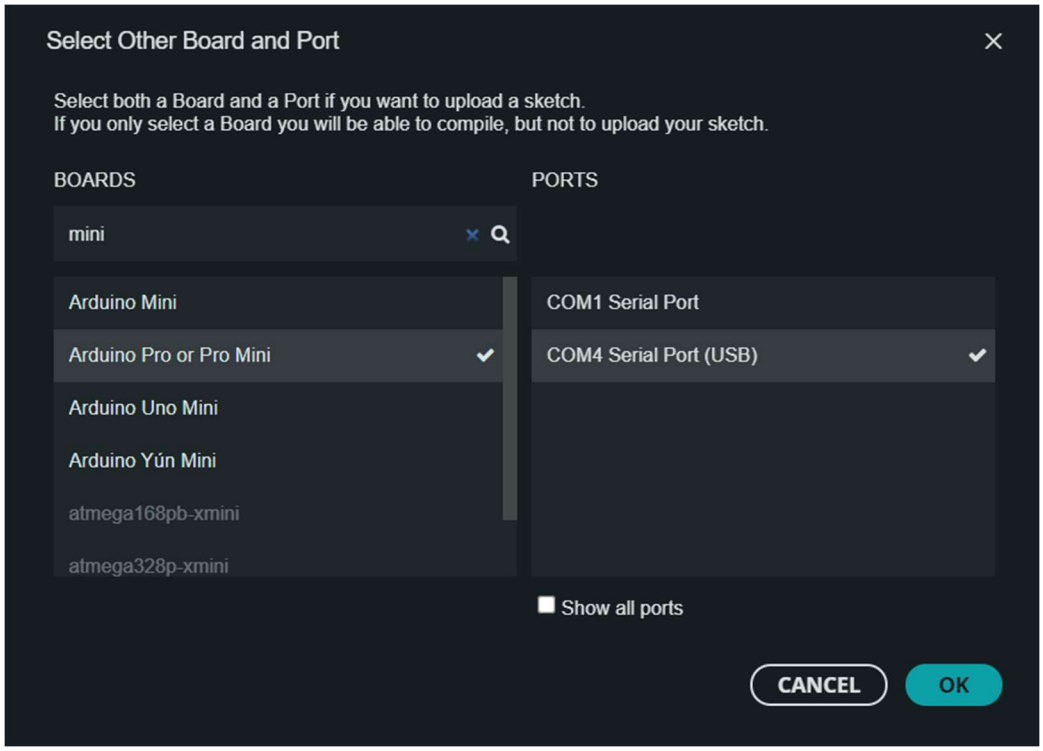
The code is available on github at the following repository:
https://github.com/rscafidi/CSUN_Lagrangian_Water_Sampler

Setting up Arduino IDE and loading the Pump Timer sketch

1. Open the Pump_Timer_MOSFET_Sleeper.ino sketch within the Arduino IDE.
    a. Launch Arduino IDE, then go to File > Open > navigate to the .ino file
2. Set the IDE to the controller model and processor.
    a. Connect the microcontroller to your PC with the USB header pin connector.
    b. Click the COM port selector dropdown

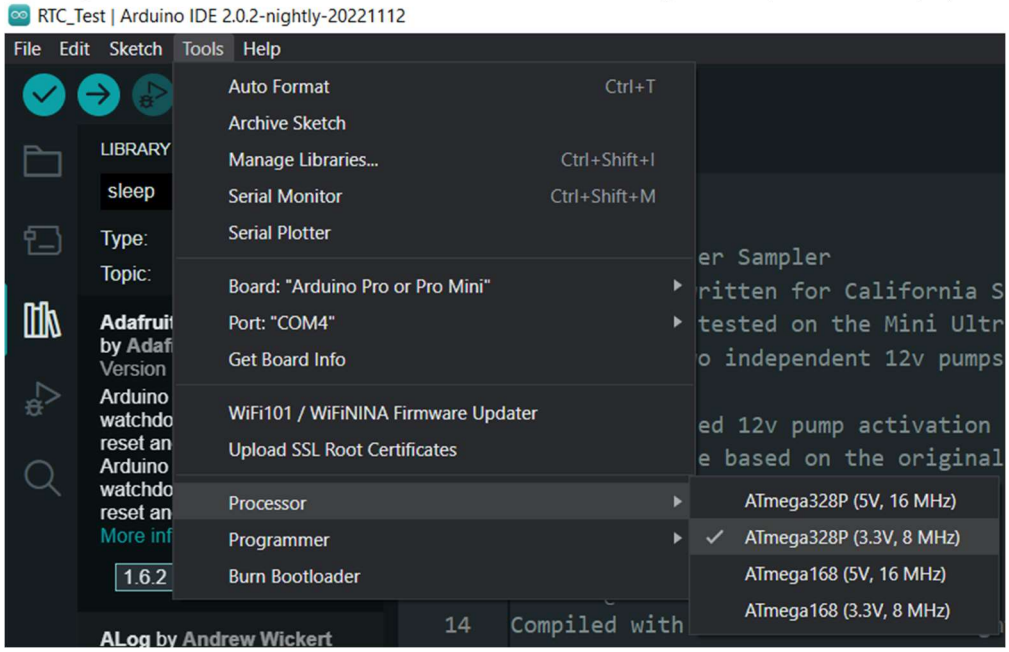


3. Choose “Select Other Board and Port” if the “Arduino Pro or Pro Mini” is not selected.

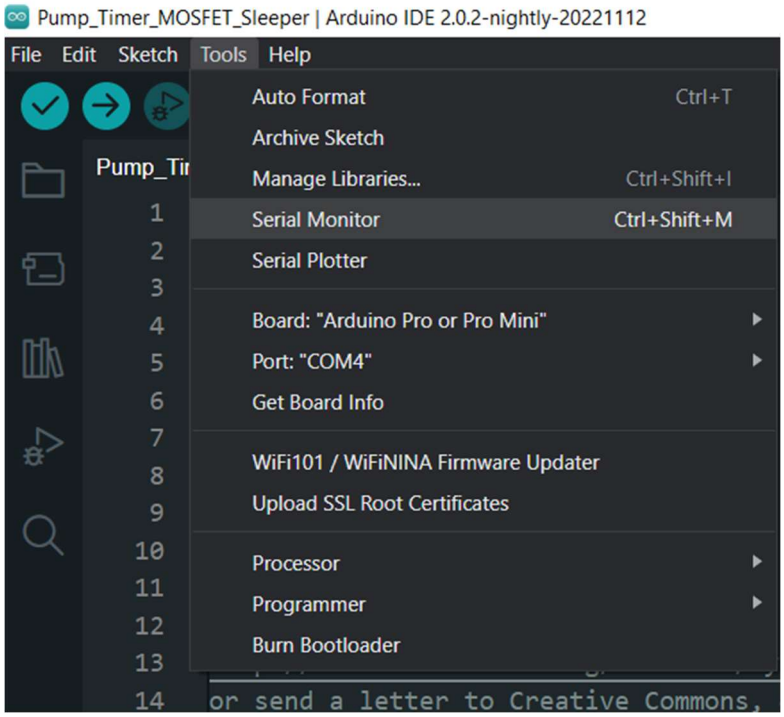


   a. Select the COM port that the microcontroller is connected to.

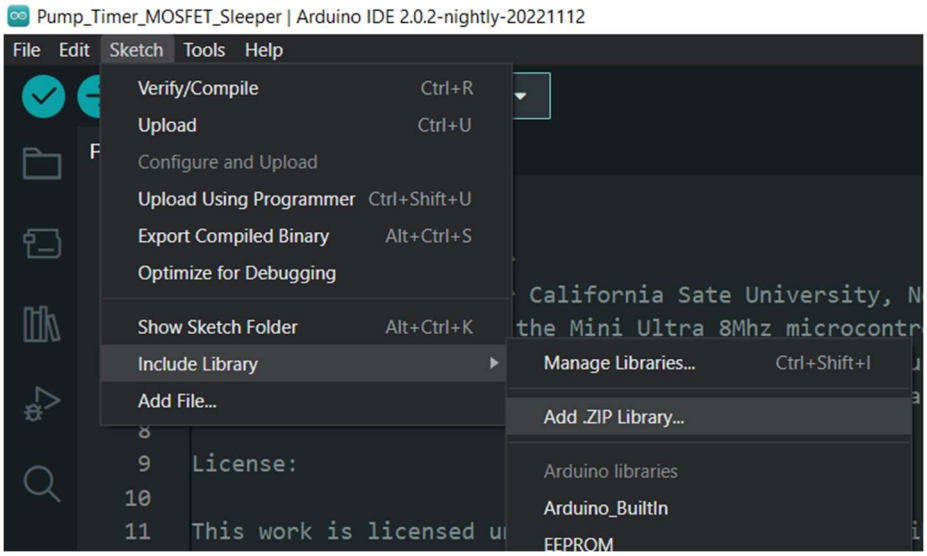
4. Next, go to Tools > Processor and select the “ATmega328P (3.3V, 8 Mhz)” processor.



5. Open the Serial Monitor by going to Tools > Serial Monitor

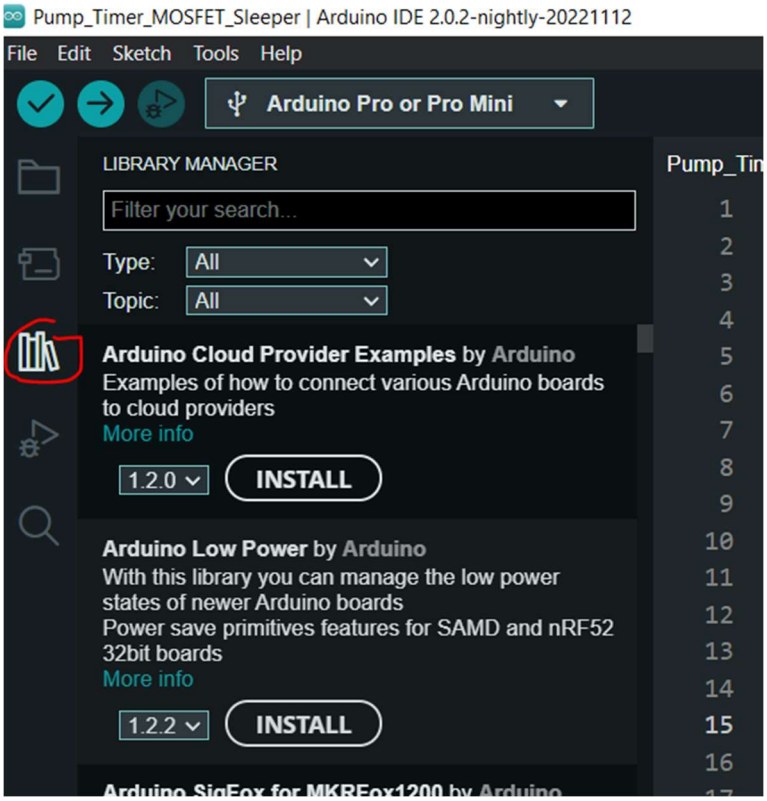


6. Add the dependency libraries.
    a. Go to Sketch > Include Library > Add .ZIP library

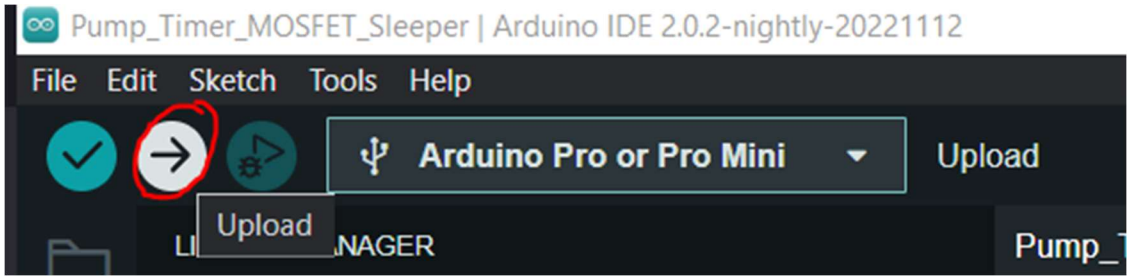


    b. Browse to the Adafruit_SleepyDog_Library.zip and add it, then do the same for RTClib.zip

c. Note you can also install libraries via the library manager within the Arduino IDE but they were included in case dependencies change in the future.

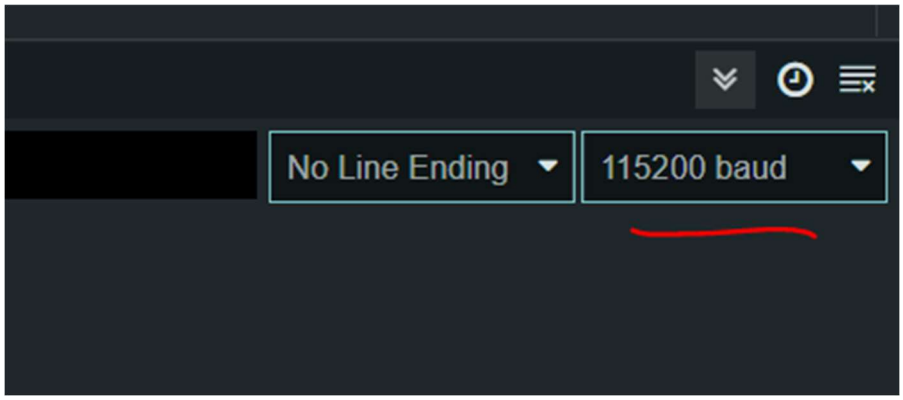


7. Put the controller into CODE_DEBUG_MODE by uncommenting line 36.
8. Click the Upload button to compile and load the program onto the microcontroller.

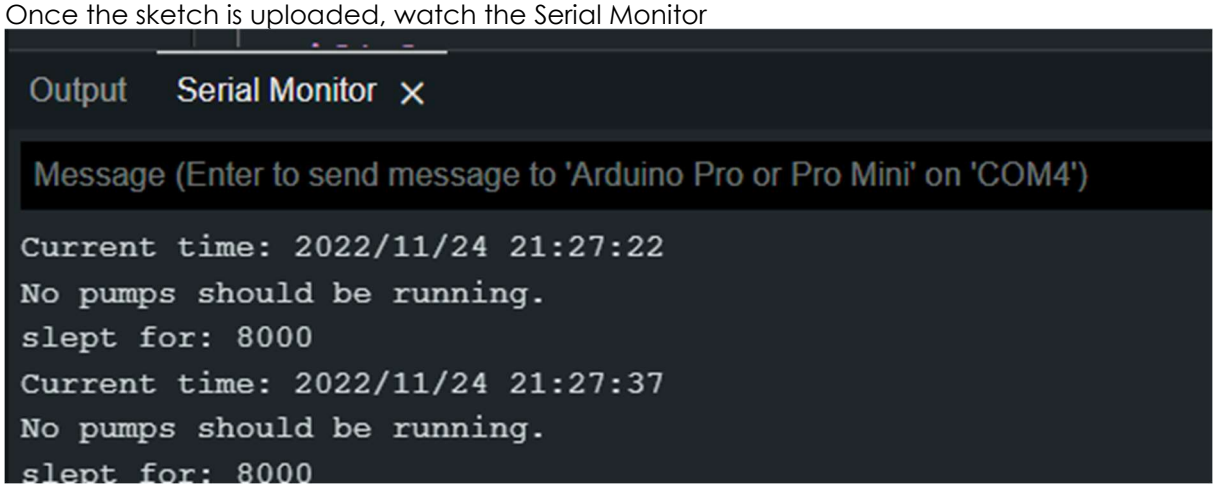


9. The Output tab will give you status on the compilation and upload. Note that if you see "Access is denied" on your COM port, try uploading again. Sometimes the COM port is busy and won't let the PC talk over it temporarily. It could also be giving you that error if you have the wrong COM port selected.

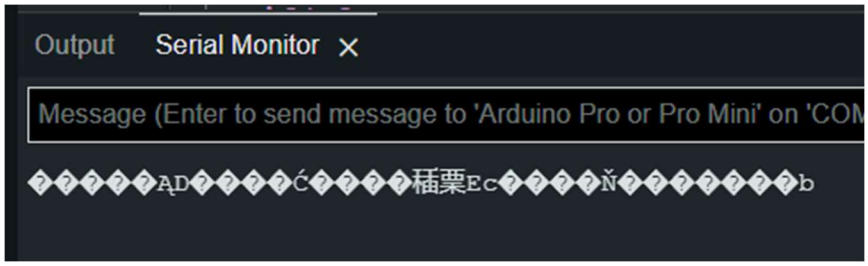
10. In the Serial Monitor window (tab next to Output on the bottom of the IDE), select 115200 for the BAUD rate on the right side.



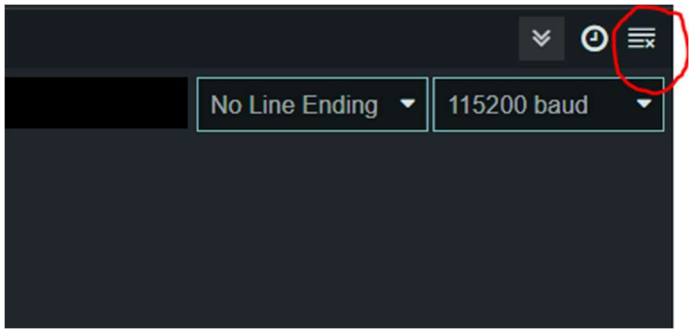
11. Once the sketch is uploaded, watch the Serial Monitor



12. If you see gibberish characters in the serial monitor window like so, you probably have the wrong processor or BAUD rate selected.



13. To clear the Serial monitor, click the Clear Output button on the top right of the Serial Monitor window.



LED LIGHT STATUS CODES

The LED light connected to pin 12 will flash in different patterns indicating various conditions of the controller.

LED Pattern	Meaning
Rapid blinking, not stopping	The microcontroller has encountered an error state, or the clock oscillator has been interrupted.
5 sets of 3 bursts	The microcontroller is in circuit debugging mode or clock sync mode. Program execution will not continue in these modes. Code must be recompiled and flashed with the control variables commented out.
20 bursts	The pumps are reporting as on during the scheduled hours they should be inactive. Troubleshooting required.
Single blink every ~10 minutes (seemingly no blink)	The microcontroller is operating normally and the pumps are both off.
Single blink every ~5 seconds	The microcontroller is operating normally and one of the pumps is active.

SWITCHING CONTROLLER MODES

The controller has three non-standard modes programmed that are controlled by commenting and uncommenting the associated control variables, starting on line 30. The first mode (CODE_DEBUG_MODE) will compile the code to produce Serial output strings that can be helpful for debugging so you can view the Serial Monitor for information. The second mode (CIRCUIT_DEBUG_MODE) will compile the code such that pump 1 and 2 will alternate being on for 5 seconds each. This can be useful for troubleshooting during times that the pumps won't normally run. The third mode (CLOCK_SYNC_MODE) is used when the DS3231 real time clock has encountered a power interruption and needs to be resynced. This mode will reset the flags indicating power interruption occurred and will sync the time with the date and time that the application was last programmed. Note that for CIRCUIT_DEBUG and CLOCK_SYNC modes, it is critical to recompile the code with these modes commented out or the normal state will never resume. CODE_DEBUG_MODE can be used during normal operation, but it should be commented out as well when you are not actively debugging things.

CHANGING THE SCHEDULE BEHAVIOR

The original requirements specified that pump 1 should run from 10AM until 4PM, and pump 2 should run from 10PM until 4AM. The current program will accomplish this schedule. It is possible to change the schedule, but it may require some slight re-programming of the time control logic.

Constants used for the start and stop times for each pump are defined on lines 74 to 78. Shifting the hours is relatively easy. However, if you must add minutes to the schedule, you will have to evaluate the pump control logic and add in the checks for minutes. The current minute is accessible using now.minute() similar to how the hour was retrieved. The control logic is defined on lines 455 to 480.

With the current control logic, the pumps will activate within 10 minutes of the appropriate hour changing. It is possible to reach a higher degree of accuracy if required but it will take more power.

CHANGING BOARD SLEEP TIME

The pump activation will be plus or minus the setting for sleep minutes (constant variable SLEEP_SECONDS defined on line 79). This equates to 10 minutes, which appears to be the previous behavior based on previous code. This means that, in the worst case, the pump could activate up to 10 minutes past the start time. If higher accuracy is desired, simply reduce the SLEEP_SECONDS variable and recompile the code. Note that lower sleep time will increase power consumption.

TROUBLESHOOTING

1. Edit code to uncomment line 45, to enable CIRCUIT_DEBUG_MODE.
2. Compile and upload the code to the microcontroller.
3. Ensure a 12v power source is connected to the MOSFET transistors.
4. Verify the 5 bursts of 3 flashes on the LED, indicating the controller is in a special control mode.
    a. Each pump will cycle ON for 5 seconds before repeating the flash cycle. Check that each pump operates.
    b. If the LED never illuminates, check the board power on the VIN pin, a minimum of 3.3v is required.
    c. If the board is powered and the code successfully uploaded but there are no LED flashes, check the signal on pin 12. If pin 12 never goes HIGH, there may be a problem with the microcontroller.
5. If the pumps are not connected, use a multi-meter to check the voltage on the output side of the MOSFET transistor. Place one lead on GROUND and place the other on the positive side of the MOSFET transistor output. Wait for the controller to cycle, a minimum of 7 seconds if it is not active when you first check.

    a. If the output never reads 12v, trace the signal from pin 3 (or 5). If the voltage is cycling on the pin wire but the output never cycles to 12v, there is a problem with the MOSFET transistor, and it should be replaced.
6. Repeat for the second MOSFET transistor.
7. Check the DS3231 onboard LED. It should be illuminated when the board is powered.
    a. If desired, put the board into CODE_DEBUG_MODE, compile and upload, and watch the serial output to ensure the time is correct.
    b. If time needs to be corrected, uncomment the CLOCK_SYNC_MODE control variable, and flash the code again.
        i. IMPORTANT: You must comment out the CLOCK_SYNC_MODE control variable and flash the code again before the microcontroller boots again. When CLOCK_SYNC_MODE is active, it will always use the last compiled time to set the real time clock. If you wait an hour and the board reboots, the clock will now be off by one hour.
8. Assuming you have been testing while the microcontroller is connected to the PC, restore the normal operational mode by commenting out all control variables and flashing the code again. Remove the controller from the USB/FTDI header and connect to 3.6v battery supply.
    a. Verify the LED is flashing once every 15 seconds if the time is outside of pump operational hours, or once per 5 seconds if the time is within pump operational hours.
9. Ensure the switch works to turn the main board off.
10. Turn the main board back on. Ensure the LED pattern continues to read a normal state.
11. If the LED ever enters a quick pulse that does not stop, an error state has been encountered.
    a. Most likely, the real time clock has lost power.
    b. Check the 3v battery supply, replace if necessary.
    c. Flash code again with CLOCK_SYNC_MODE and CODE_DEBUG_MODE uncommented.
    d. Watch the serial output and make sure it goes back to a normal state.
    e. Comment the special control variables and flash code, check LED again and power cycle the board.
12. If the board is properly resetting the clock registers (success messages when board is in CLOCK_SYNC_MODE) but the real time clock is losing power in between board cycles with a known good 3v battery, this could indicate a problem with the real time clock. Recommend replacing the module.

APPENDIX D – BILL OF MATERIALS

Below are the items that comprise the Lagrangian Water Sampler. These parts were provided by CSUN and in some cases, replacements were ordered. This list includes each component with applicable part numbers where possible.

Item Friendly Name	Part Number or ID	Functional Description
Mini Ultra 8Mhz Microcontroller	ATMega328P 8Mhz 3.3v	Microcontroller for executing code to control LED and pumps
MOSFET Transistor x2	FQP30N06L	Transistor to open circuit to 12V power in order to power the pumps
Real Time Clock module	DS3231	Real time clock with battery power supply to retain time in between board power cycles
LED	Standard	Light emitting diode to be used for status indications
FTDI USB cable	FTDI USB to TTL Serial Female Socket Header	Connect microcontroller to computer with Arduino IDE, for flashing new code to the controller
Watch battery	2032 3v battery	For powering the RTC module
Standard electronics wiring	22 AWG electronic wiring	For connecting components
Board power supply	3.6v Li battery	For powering the microcontroller
Welco water pump x2	WP10 Peristaltic Pump	For pumping water
12v Power Supply	-	For powering pumps
Bread Board	-	For connecting and soldering all components
10k Ohm Resistors	-	Wired with transistor
Custom Plexiglass Case	-	Housing for control unit
D Cell size battery holder	-	For holding the board power supply.

Sources:

https://smile.amazon.com/gp/product/B004UHUGGC/ref=ppx_od_dt_b_asin_title_s00?ie=UTF8&psc=1

https://smile.amazon.com/gp/product/B07PKKY8BX/ref=ppx_yo_dt_b_asin_title_o01_s00?ie=UTF8&psc=1

https://smile.amazon.com/gp/product/B073QMYKDM/ref=ppx_yo_dt_b_asin_title_o01_s01?ie=UTF8&psc=1

https://smile.amazon.com/gp/product/B08B8WRQP1/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&psc=1

https://smile.amazon.com/dp/B07Q7NZTQS?psc=1&ref=ppx_yo2ov_dt_b_product_details

https://www.rocketscream.com/blog/product/mini-ultra-8-mhz-arduino-compatible/