Final Project Report

For mini-Bluetooth Project

Leonardo Fusser, 1946995

Project presentation done on **25 January 2021**
Report Submitted on **26 May 2021**

Department of Computer Engineering Technology
Telecommunications (247-410-VA)
Dr. Manijeh Khataie

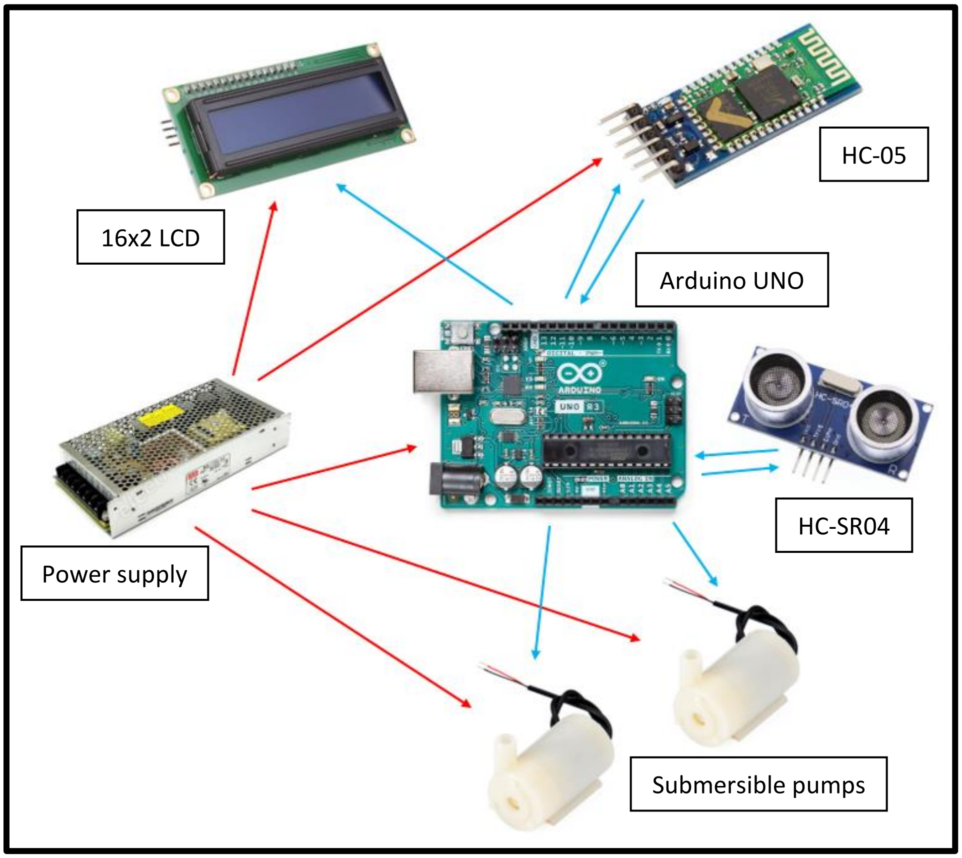
VANIER
CÉGEP/COLLEGE
Learning today Leading tomorrow

TABLE OF CONTENTS

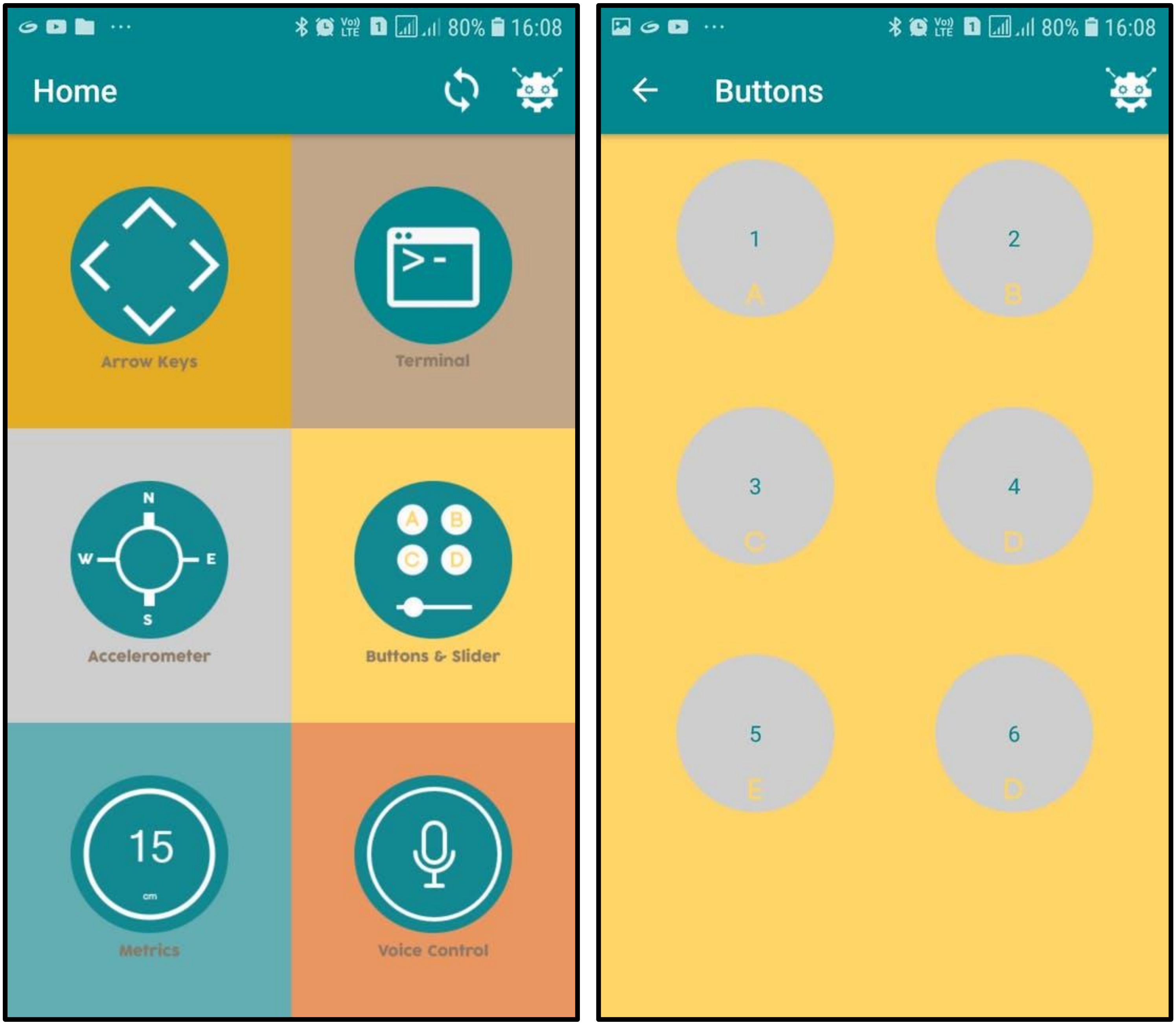
1.0 Introduction

Overview of mini-Bluetooth project

- This mini-Bluetooth project is a remote-controlled water dispensing system using Bluetooth 2.0 technology to control the project. A HC-05 Bluetooth module is used to control the project and the Arduino BlueControl app, installed on any Android device (Bluetooth module not compatible with iOS), is used to communicate with the Bluetooth module (see next page for view of how interface looks like). The project consists of two tupperwares (water holding tanks), each with its own submersible pump that has its outlet going to the opposing tupperware so water can be filled in there. A HC-SR04 ultrasonic sensor is attached to one of the tupperwares and is used to measure the level of the water in its corresponding tupperware. Based on the reading taken by the ultrasonic sensor, a message will be displayed on a 16x2 LCD display indicating which tupperware is full and which one is not. This display provides additional user interface on top of the user interface provided by the Arduino BlueControl app installed on any Android device to the user. At the heart of this system is an Arduino UNO which handles all project behavior. It has the Bluetooth module interfacing to it as well as the LCD display, ultrasonic sensor and both submersible pumps. All components of this project run on 5V input power. A basic diagram of the project can be found below.



Basic diagram of mini-Bluetooth project shown above.

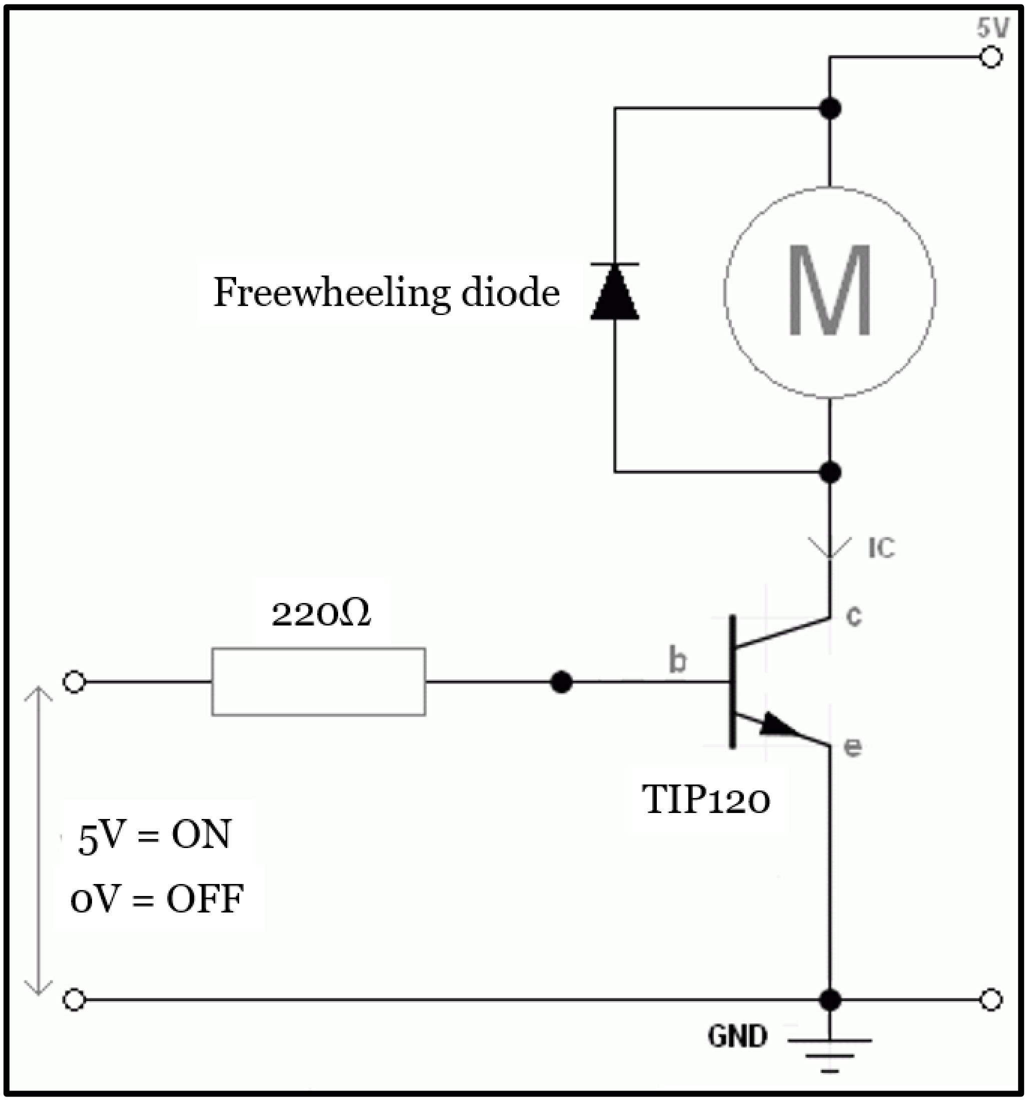


View of how Arduino BlueControl interface looks like on any Android mobile device shown above.

2.0 Project Assembly

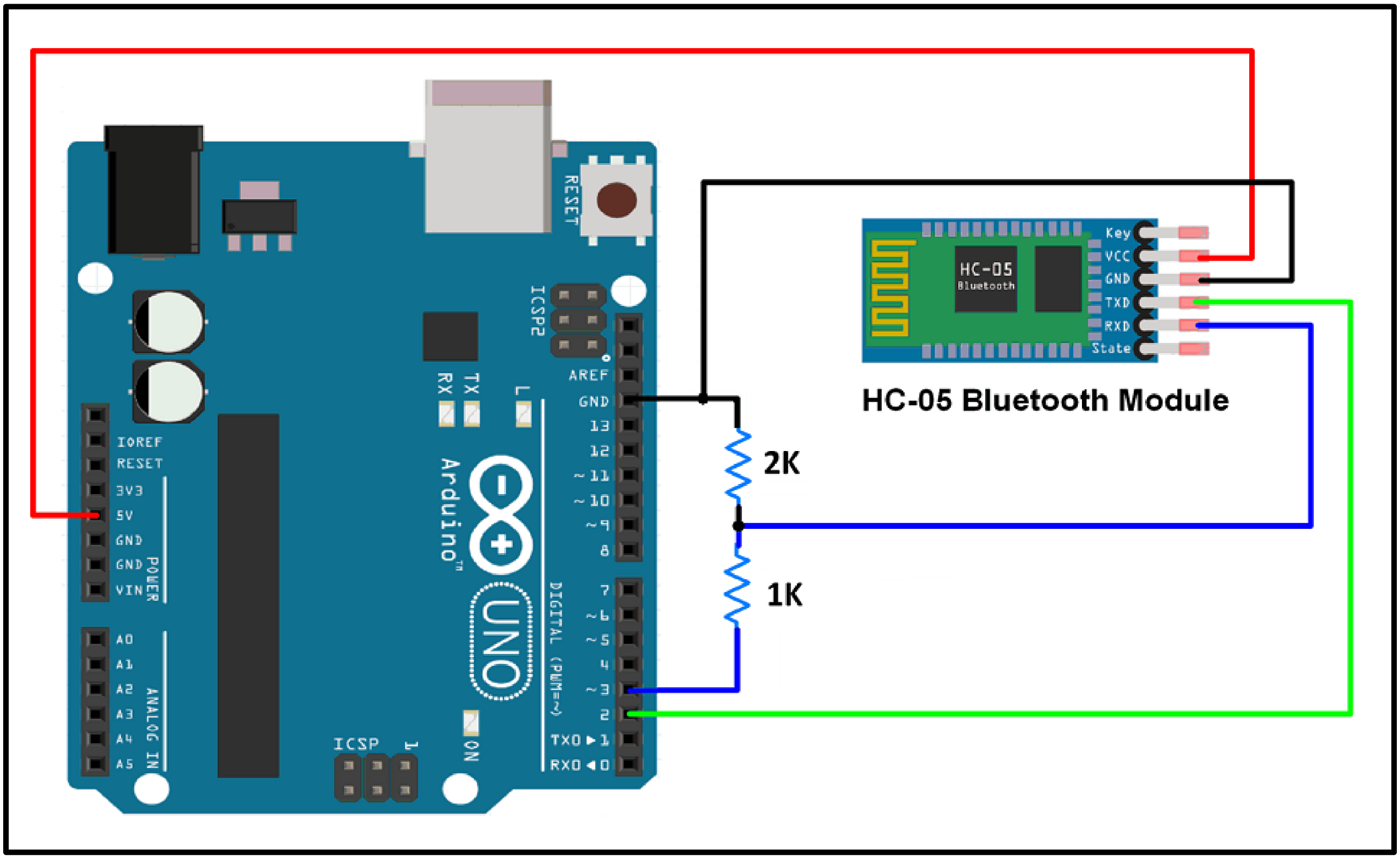
Putting everything together

- Once a basic diagram was created for the mini-Bluetooth project, assembly of the project was able to begin. It is important to note that during this phase of the project, basic prototyping was done in order to ensure proper functionality and that what is shown on the next few pages is the implementation of the best results from it. The first thing that was put together was the submersible pump assembly and can be shown in the schematic below.



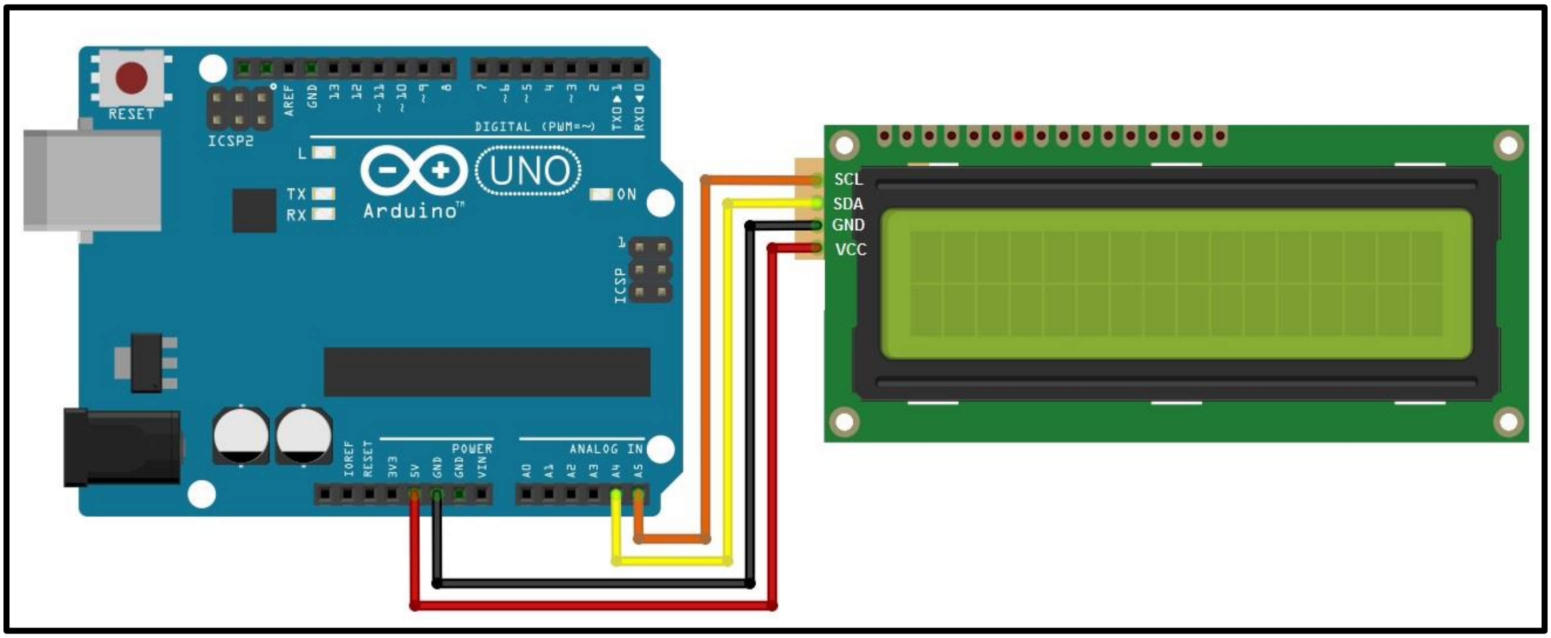
Schematic of submersible pump assembly shown above.

- Each of the two submersible pumps has a circuit wired as shown above. This is a basic BJT circuit, using a TIP120 NPN transistor, configured to act like an "on" and "off" switch. A freewheeling diode is in place for protection and a 220Ω is in series with a green LED (not shown above) and is applied to the base of the transistor. When +5V is supplied from the Arduino UNO, the transistor turns "on" and the submersible pump runs. When 0V is supplied from the Arduino UNO, the transistor turns "off" and the submersible pump does not run. The submersible pump is supplied power from a +5V source and is not powered from the Arduino UNO (explained later in this report why). Final pictures of the implementation of the submersible pumps will be shown in the next few pages. Once the submersible pumps were implemented, the next step was to wire the HC-05 Bluetooth module to the Arduino UNO and a schematic of this assembly can be found below.



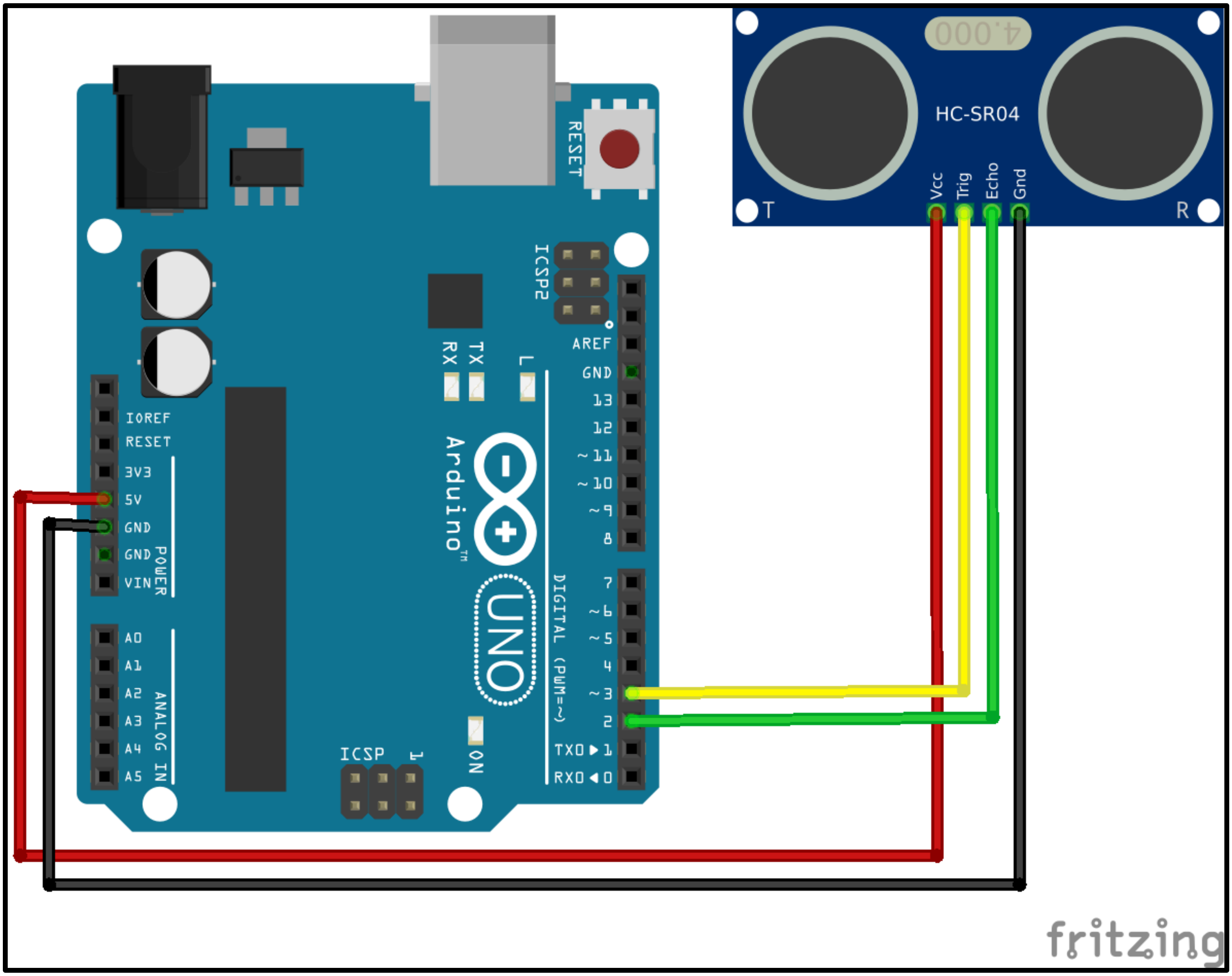
Schematic of HC-05 Bluetooth module assembly shown above.

- The HC-05 Bluetooth module is supplied with +5V source power but can only tolerate +3.3V inputs. Since the outputs of the Arduino UNO are +5V tolerant, a voltage-divider circuit is needed in order to protect the inputs of the HC-05 Bluetooth module. A 2kΩ resistor in series with a 1kΩ deemed fit to satisfy this demand. The only difference from what is shown in the schematic above is that the TX0 and RX0 pins were used on the Arduino UNO. TX0 pin (from Arduino UNO) goes to RX pin through voltage-divider circuit on HC-05 Bluetooth module. RX0 pin (from Arduino UNO) goes to TX pin on HC-05 Bluetooth module. A 10uF capacitor was also added between the +5V and GND lines for filtering as the HC-05 Bluetooth module was experiencing some weird behavior. Final pictures of the implementation of the HC-05 Bluetooth module will be shown in the next few pages. Once the HC-05 Bluetooth module was implemented, the next step was to wire the 16x2 LCD display to the Arduino UNO and a schematic of this assembly can be found below.



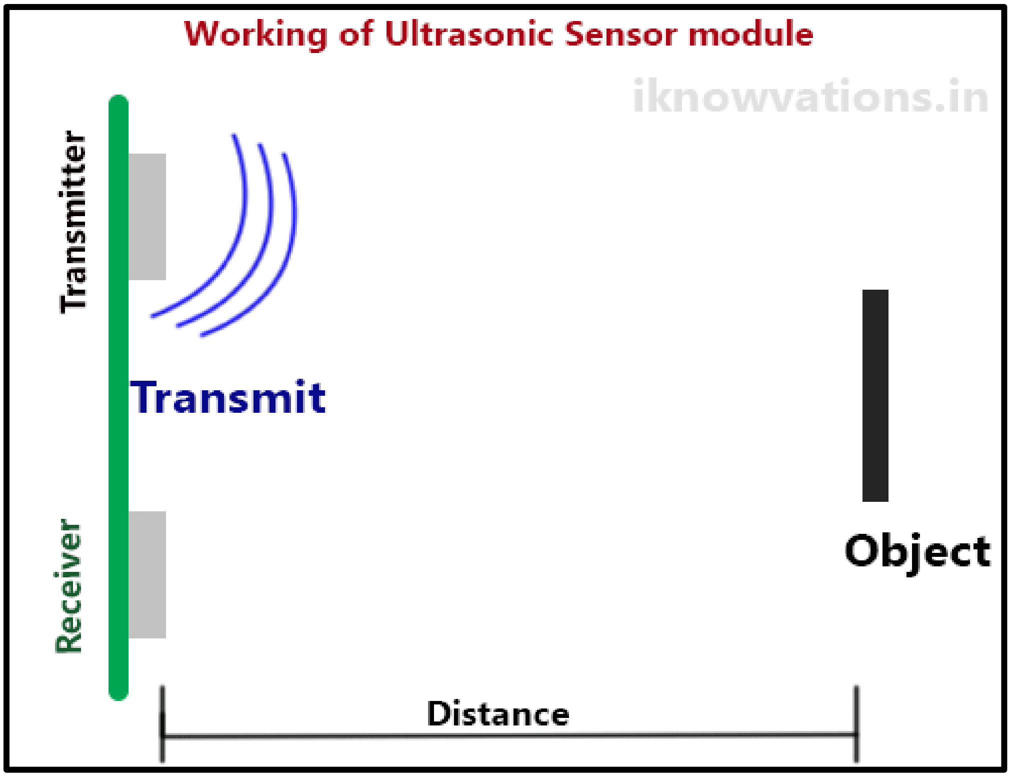
Schematic of 16x2 LCD assembly shown above.

- The 16x2 LCD display is supplied with +5V source power but can only interface to analog pins on the Arduino UNO. This display has an i2c module directly connected on the back of the display (not shown above) and can only interface to analog pins on the Arduino. As this may not be a good thing for some, it does reduce the number of pins taken up on any microcontroller to drive the display. The inputs of the i2c module interface to two specific pins analog pins on the Arduino UNO. The SCL pin interfaces to the A5 pin and the SDA pin interfaces to the A4 pin on the Arduino UNO. Depending on the Arduino being used, this can differ. Final pictures of the implementation of the 16x2 LCD display will be shown in the next few pages. Once the 16x2 LCD display assembly was implemented, the next step was to wire the HC-SR04 ultrasonic sensor to the Arduino UNO and a schematic of this assembly can be found on the next page.

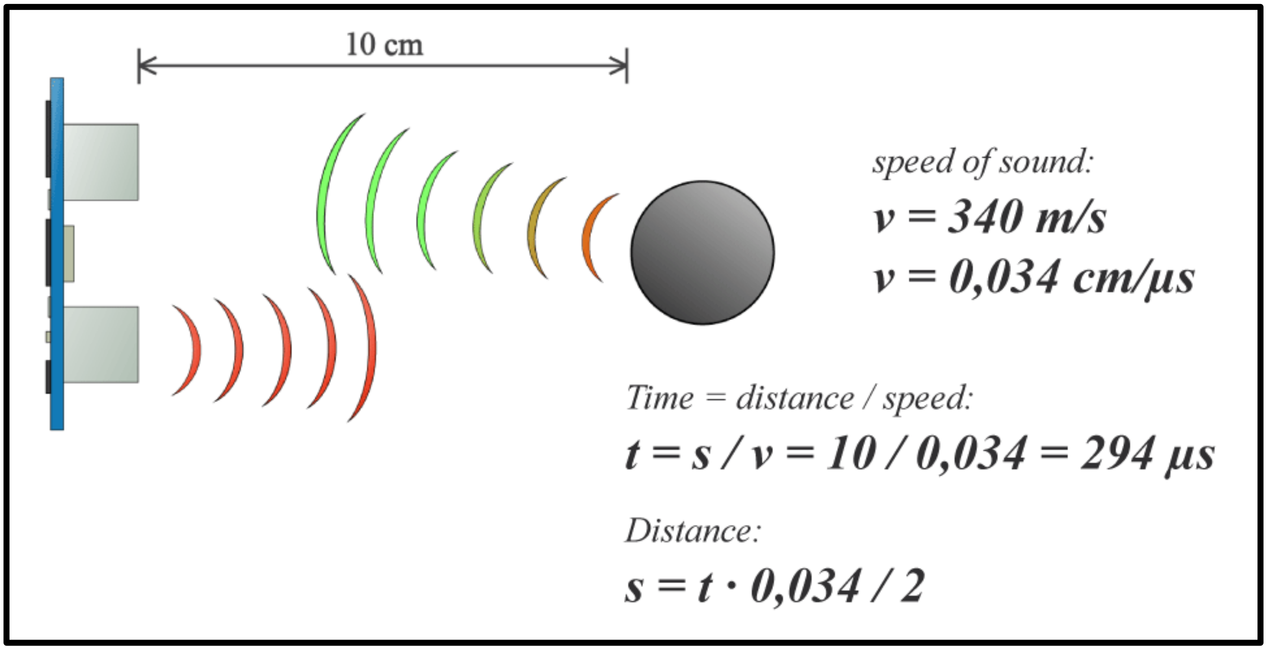


Schematic of HC-SR04 ultrasonic sensor assembly shown above.

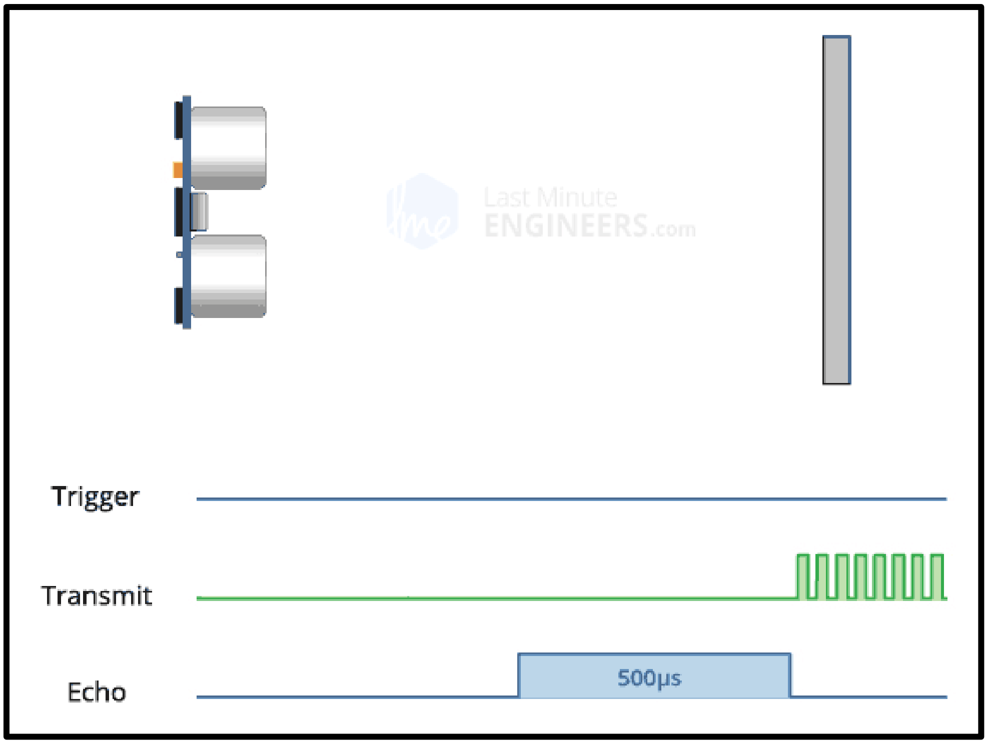
- The HC-SR04 ultrasonic sensor is supplied with +5V source power and unlike with the 16x2 LCD display, it can interface to any of the pins on the Arduino UNO. Similar to the 16x2 LCD display, it only takes up two pins from the Arduino UNO. The “Trig” pin interfaces to pin D3 and the “Echo” pin interfaces to pin D2 on the Arduino UNO. There is some specific logic behind the functionality of this ultrasonic sensor and the Arduino UNO needs to be programmed in a specific way in order for this ultrasonic sensor to work correctly. A quick rundown of how this ultrasonic sensor works can be found in the next two pages.



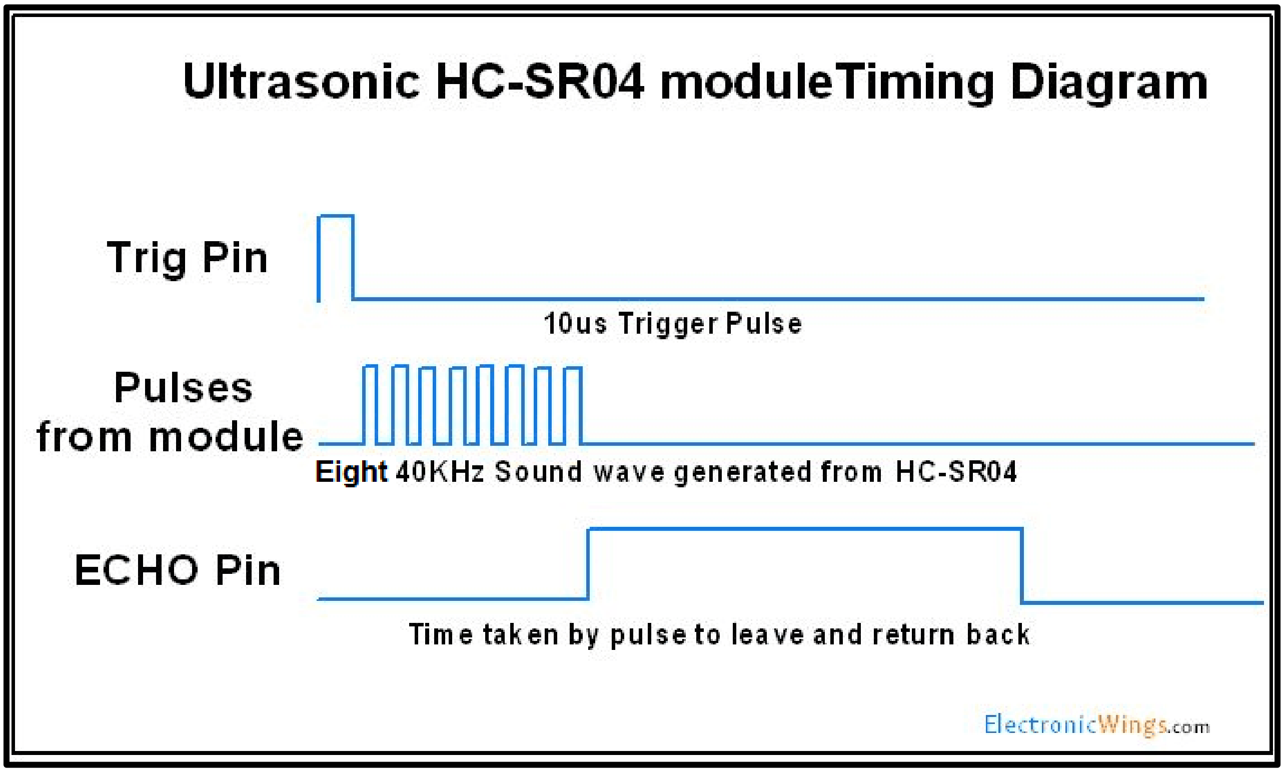
Animation of HC-SR04 working in real-time shown above.



Math implemented in programming to properly determine distance ultrasonic sensor reads shown above. Some prototyping was done with the ultrasonic sensor to determine the full and low readings for the tupperwares holding the water.

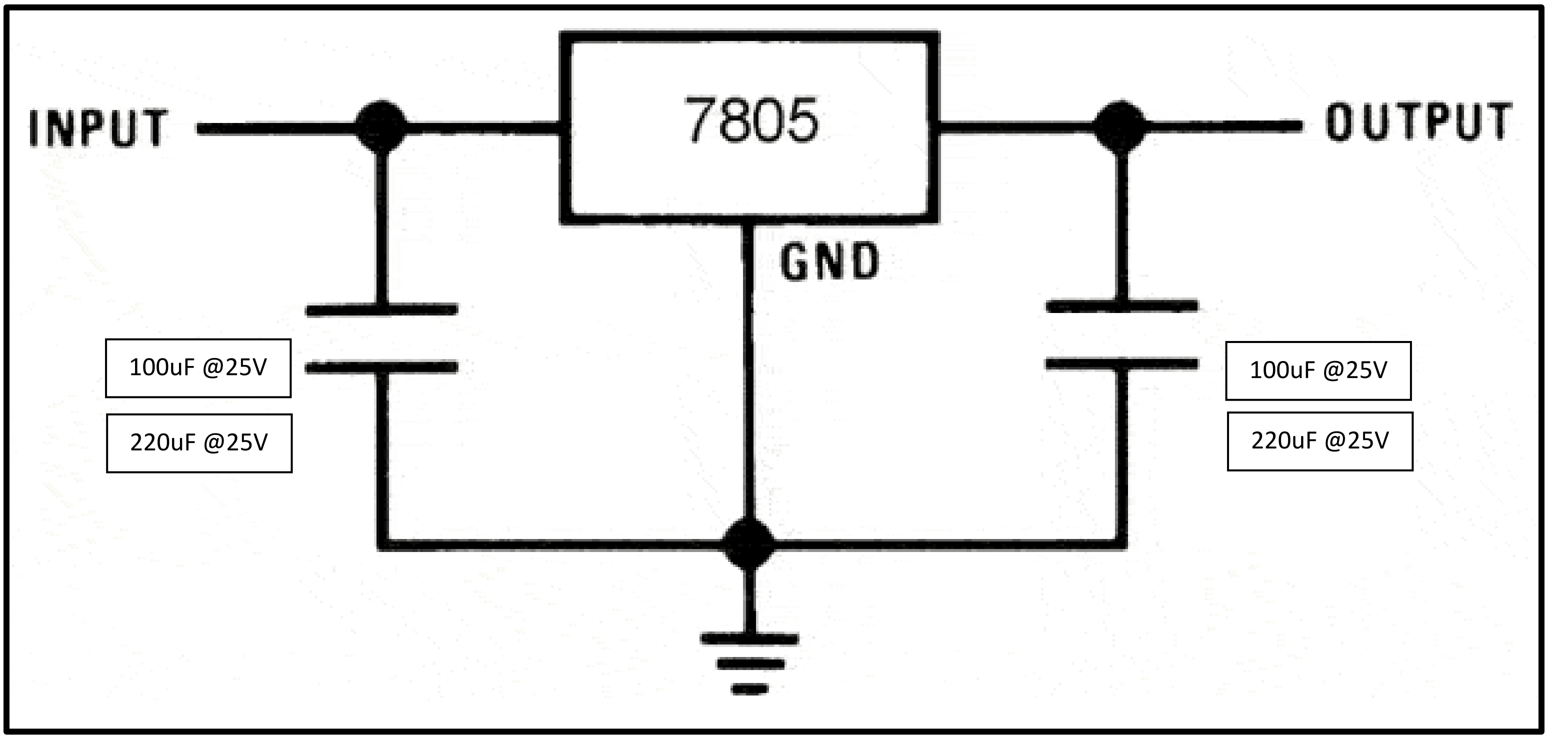


Another animation of HC-SR04 working in real-time with timing diagram shown above.


Ultrasonic HC-SR04 moduleTiming Diagram
Trig Pin
10us Trigger Pulse
Pulses from module
Eight 40KHz Sound wave generated from HC-SR04
ECHO Pin
Time taken by pulse to leave and return back
ElectronicWings.com

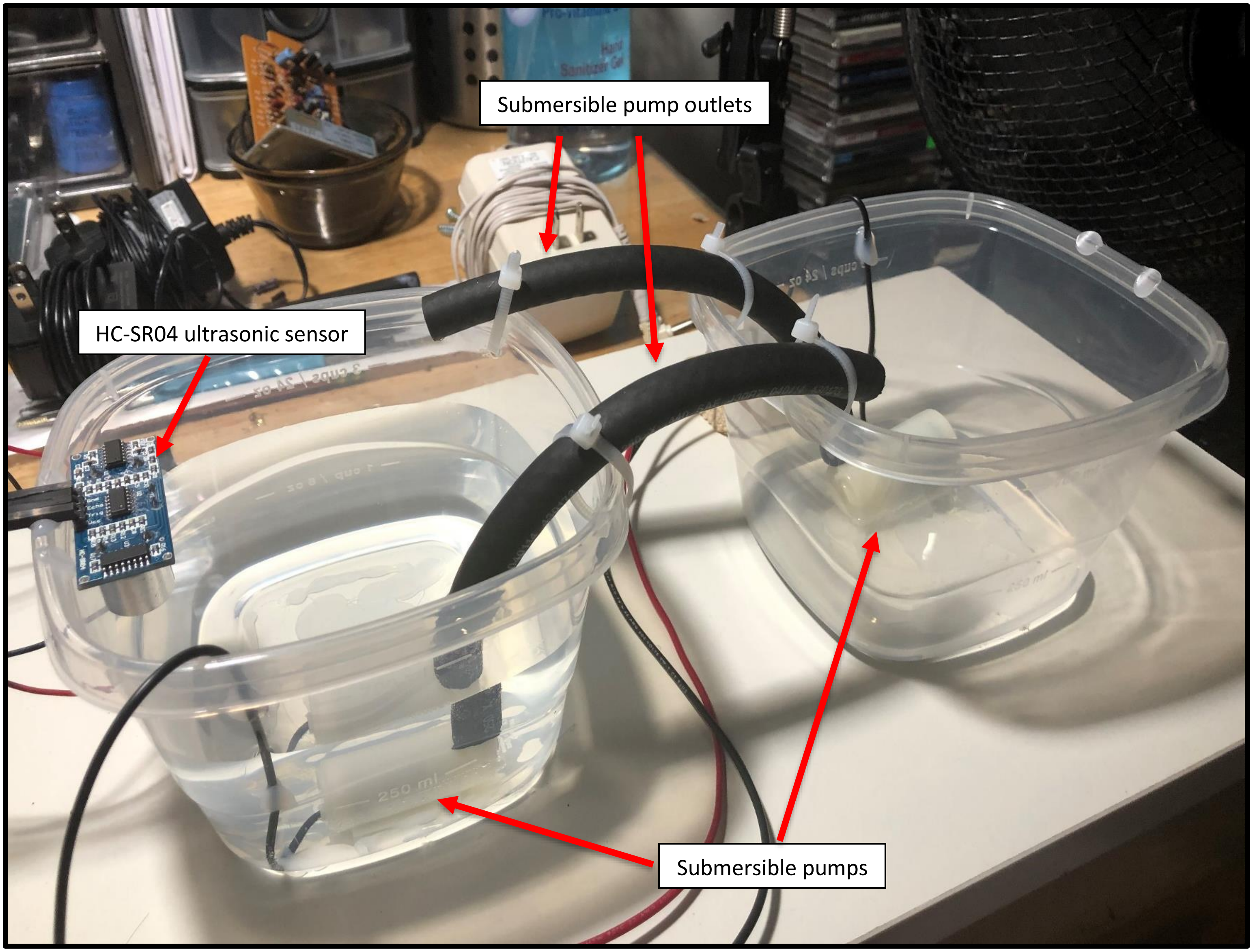
Timing diagram for HC-SR04 ultrasonic sensor shown above.

- The last step of the project assembly was to wire the two regulated +5V circuits that power the all the peripherals of the project. Power for the Arduino UNO, 16x2 LCD display, HC-05 Bluetooth module and HC-SR04 ultrasonic sensor were powered off one of the two regulated +5V circuits. The other regulated +5V circuit supplied power to the two BJT circuits powering the two submersible pumps that are in the two tupperwares. A schematic of this assembly can be found in the in the picture below.

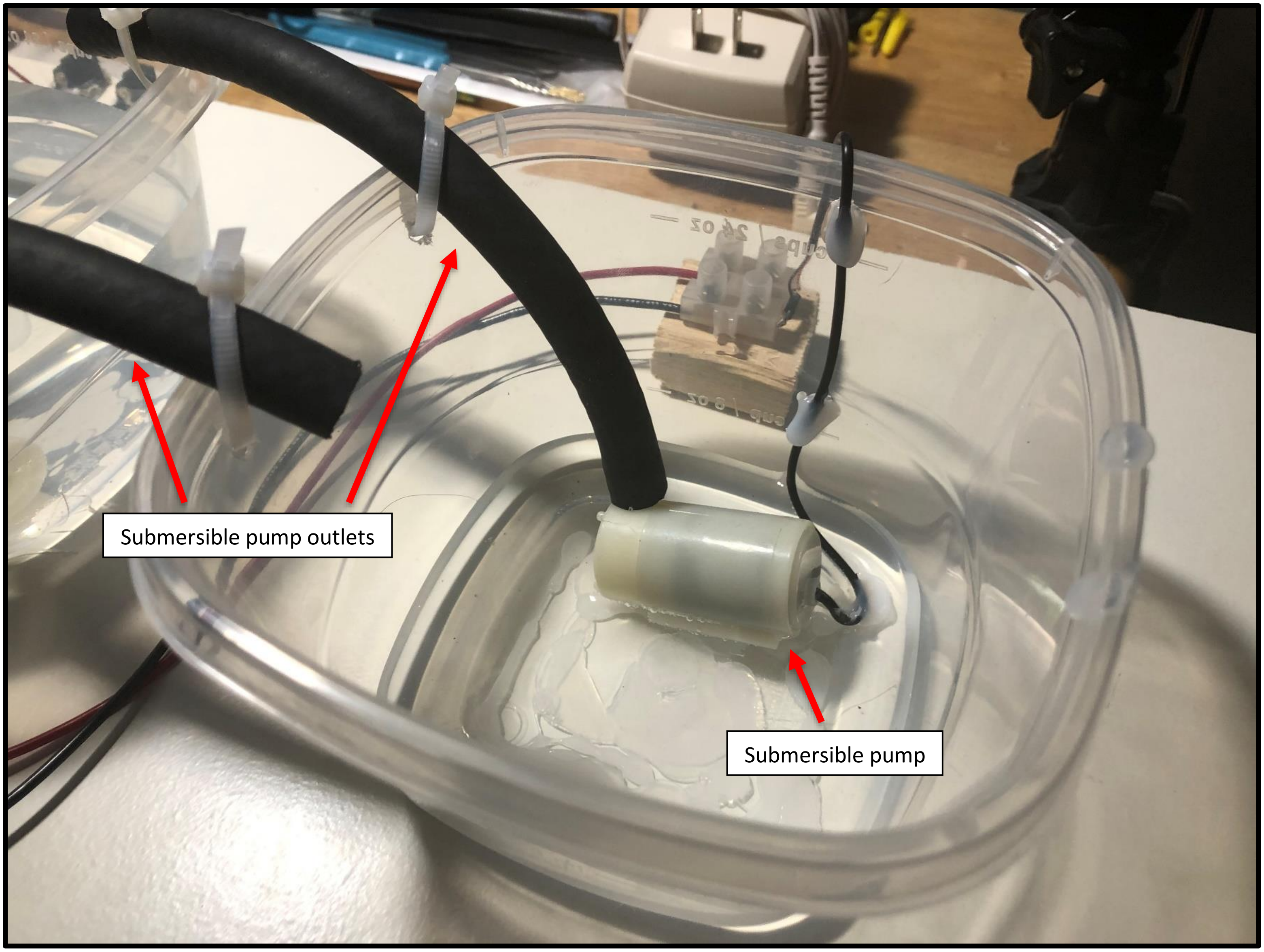


Schematic of both regulated +5V assemblies shown above.

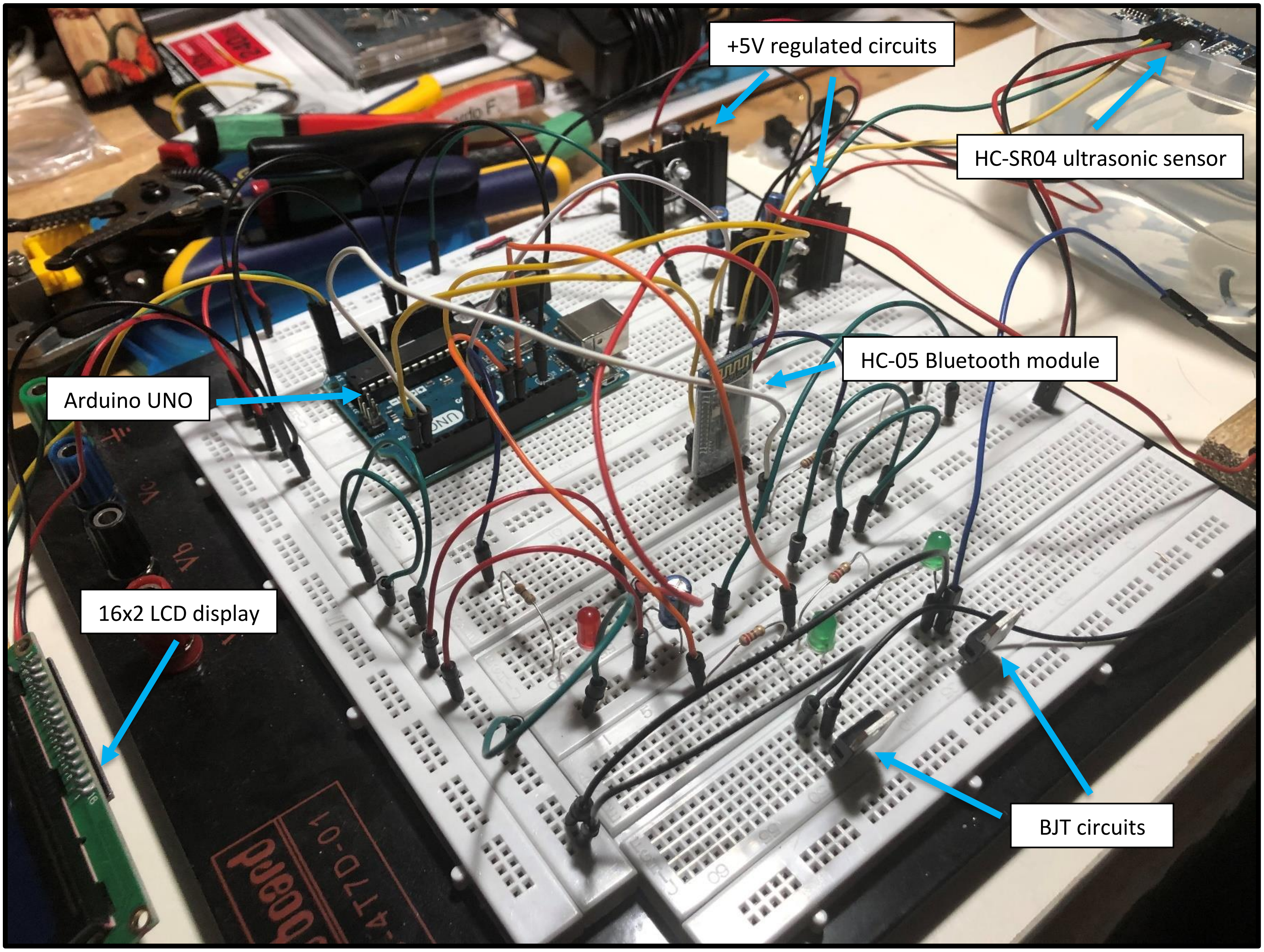
- Both of the regulated +5V circuits are based off the 7805 linear regulators. For the regulated +5V circuit powering the Arduino UNO, HC-05 Bluetooth module, 16x2 LCD display and HC-SR04 ultrasonic sensor, a pair of 220uF @ 25V capacitors were used. The other regulated +5V circuit powering the two BJT circuits powering the two submersible pumps that are in the two tupperwares used a pair of 100uF @ 25V capacitors. For both of the regulated +5V circuits, a 12V 2A power supply feeding to the inputs of the 7805s was used. What is not shown above is that heatsinks were attached to the 7805s in order to maintain proper functionality (will be explained why later in this report). Final pictures of the assembly of the mini-Bluetooth project are shown in the next few pages.



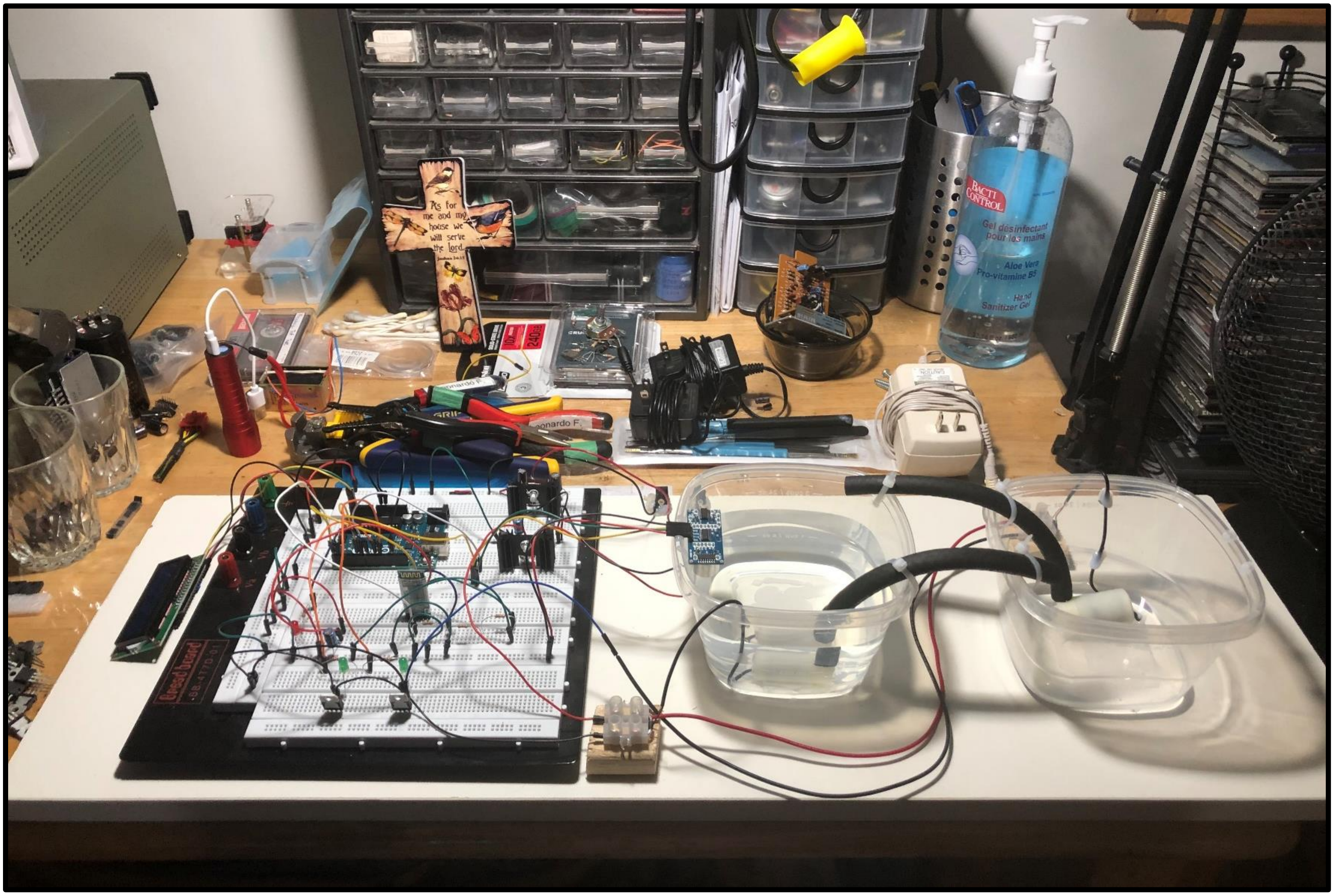
Final assembly of the two tupperwares with the two submersible pumps in each one shown above.



Final assembly of the two tupperwares with the two submersible pumps in each one shown above.



Final breadboarding and wiring of entire mini-Bluetooth project shown above.

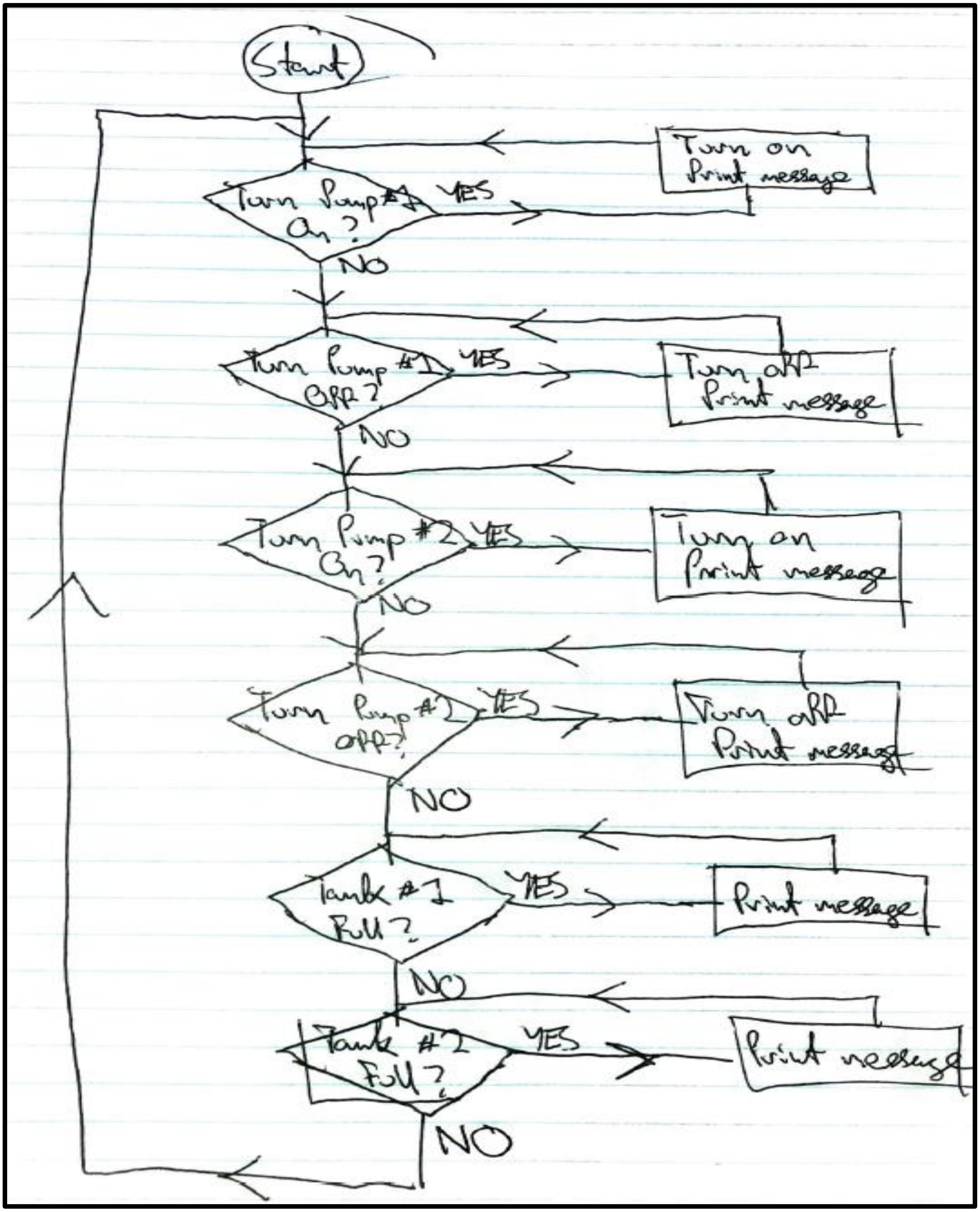


Final assembly of entire mini-Bluetooth project shown above. Entire project is secured to a piece of MDF board for easy transportation and demonstration of project.

3.0 PROJECT PROGRAMMING

Programming the Arduino UNO

- Once all physical and electrical assemblies were complete, the last step in order to make this mini-Bluetooth project functional was to program the Arduino UNO, as it is the only microcontroller that needs to be programmed in this project. The Arduino UNO needed to be able to handle the data being sent from the HC-05 Bluetooth module to itself (interpret data using ASCI table). No direct programming was needed for the HC-05 Bluetooth module. Based on the data being sent from the HC-05 Bluetooth module, the Arduino UNO had to send data to the 16x2 LCD display to print a specific message (water level in both tupperwares) and either keep one of the submersible pumps on or off. The Arduino UNO constantly kept record of the water level that was in both tupperwares. The Arduino UNO took data from the HC-SR04 ultrasonic sensor in order to properly determine the level of water in the tupperwares (specific math was implemented here and was described previously in this report). Instead of copying and pasting the entire code used for the Arduino UNO, a flowchart on the next page describes the logic behavior that was programmed onto the Arduino UNO. Complete code can be viewed separately from this report and is attached at the same time during submission of this report.



Flowchart describing Arduino UNO logic behavior shown above.

4.0 List of Components

Final list of components used

- (1x) HC-SR04 ultrasonic sensor.
- (1x) 16x2 LCD display with i2c module.
- (1x) HC-05 Bluetooth module.
- (1x) Arduino UNO.
- (2x) 7805 linear voltage regulators.
- (2x) 12V 2A power supplies.
- (2x) 220uF @ 25V electrolytic capacitors.
- (2x) 100uF @ 25V electrolytic capacitors.
- (1x) 10uF @ 10V electrolytic capacitor.
- (3x) 1kΩ 1/4W resistors.
- (2x) 220Ω 1/4W resistors.
- (2x) TIP120 high-power NPN transistors.
- (2x) 1N007 diodes.
- (2x) double-slot terminal blocks.
- (2x) TO220 heatsinks with thermal compound and TO220 mica insulators.
- (2x) 5V submersible pumps.
- (2x) plastic tupperwares.
- (2x) ¼' rubber tubing.
- (1x) four-column breadboard.
- (?x) general purpose 22 gauge wire.
- (?x) zip-ties.
- (?x) hot glue.
- (1x) MDF board.
- (?x) appropriate tools for project assembly.

5.0 Problems Encountered & Potential Emprovements

What happened unexpectedly and what could have been done better

- Most of what was planned beforehand for the project went accordingly but some unexpected issues were encountered. Firstly, for the two +5V regulated circuits, heatsinks needed to be applied in order to prevent the 7805s from burning up. This is because the efficiency for these regulators is not so great and emit a lot of heat. The second problem that was encountered was that the submersible pumps drew too much power for the Arduino UNO to handle. Due to this, a sperate circuit that powers these two pumps was needed. The solution was to implement two

BJT circuits that acted like “on” and “off” switches which ended up working just fine. The third problem that was encountered unexpectedly was that the HC-05 Bluetooth module would constantly disconnect from my mobile device despite proper wiring and handling of it. The solution was to place a 10uF filtering capacitor between the supply lines for the HC-05 Bluetooth module. At the same time, due to the lack of compatibility with iPhones and other Apple devices, only Android devices are compatible with the HC-05 Bluetooth module so a Android device was used in order to make the project functional. Aside from these major issues, minor issues such as logic issues during programming were encountered but quickly resolved after looking at my flowchart.

- The one major improvement that I want to have like to have done was to make all the wiring for the project neater. Due to time constraints, this was not possible so quick breadboarding using jumper wires was used. As well, if time really was not an issue, I would have developed a custom PCB for the Arduino and it’s connected peripherals in order to remove the breadboard entirely. At the same time, I would have spent some more time into understanding the functionality of the HC-05 Bluetooth module as this was the first time I have used it. Most other components used in this project were not hard to implement due to my previous knowledge of their functionality.

6.0 CONCLUSIONS

Final thoughts

- Understood how Bluetooth 2.0 communication works.
- Understood how to implement and use HC-05 Bluetooth module.
- Understood how to program Arduino UNO for HC-05 Bluetooth module.
- Understood how to interface HC-05 Bluetooth module to Arduino UNO.
- Understood how use mobile phone to communicate with HC-05 Bluetooth module.
- Understood how to convert data being sent from HC-05 Bluetooth module to understandable form (thanks to ASCI table).