Project Group IC-RG-RJ

Ian Cruikshank | icruiksh@sfu.ca
Ryan Garofano | rgarofan@sfu.ca
Rajnesh Joshi | rajneshj@sfu.ca
Jaydon Vanloo | jvanloo@sfu.ca

Water Buddy Milestone 2

November 10th, 2022

Project Description

Our project is the “Water Buddy,” an embedded device that can assist users in maintaining their daily water intake by tracking the amount of water dispensed from the refrigerator, and reminding them to drink water via text message. It utilizes a weight sensor to detect the amount of water, an LCD module to display relevant information to the user, and RFID tags for each user’s water bottle so they can be uniquely identified by the device.

Accomplishment 1:

Our first accomplishment is writing a module for the RFID reader we are using to identify users. The main feature of the module is an easy to use function that attempts to get the UID of the RFID tag in front of it and returns an error status if none are there. The RFID reader communicates with SPI so we also had to write a simple SPI module to be able to handle transfers and initialize the SPI bus used. The reader module was informed by mainly the data sheet and also an Arduino library for the same model of reader. The sources of all borrowed snippets have been cited in the code where they are referenced. Our RFID reader module is quite a bit more simplistic/crude than the Arduino library, e.g. it doesn’t do all the security checks that the Arduino library does, but from our testing we’ve seen that it is very reliable.

Accomplishment 2:

Our second accomplishment is getting input from our users via a web server with a front end that includes a form. The user can fill out three fields - one for their daily water intake goal in litres, one for their phone number, and a set of radio buttons to select how often they want to be sent text message reminders. The web server is running via node.js on port 3000. A get request for ‘/’ (localhost:3000) renders an embedded javascript file which contains the form. The user fills out each field and then presses the save button. The save button stores the form data into a variable. When doing a get request for ‘/data’ the server puts the latest data stored in the form data variable into a .json file that our C program on the board can read.

Accomplishment 3:

Our third accomplishment is sending text message reminders to our users via the node.js web server. By sending a post request to the server (Ex. localhost:3000/sms/604XXXXXXX) we are able to send a reminder text message to the phone number specified in the URL. We are using the Twilio API in order to achieve this functionality.

Roadblocks:

A roadblock that we encountered was a possible faulty weight sensor that we are having some difficulty configuring.

Specifically, finding datasheets that give interpretable information for how to write to the Display RAM, and the character RAM. The display module, based on datasheets that we have read, we are aware of instructions we can use to put characters on it, but it seems to require 10 bits and doing that via I2C has been tricky.