**Project Summary**

Group 81: Adam Hurd and Julia Adamczyk

- TEAM RESPONSIBILITES -

|  |  |
| --- | --- |
| Adam Hurd | Julia Adamczyk |
| Circuit design | Interrupt testing and design |
| Hardware testing | Overall program flow |
| Flowchart/high-level design | Register-level assignments |
| LCD software |  |
| Motor software |  |
| Timer functionality |  |
|  |  |

- ISSUES & CHALLENGES –

Getting a real-time/second-based delay feature to work. Our initial thought was to use the RTC module because it’s already set up to work in real-time, but the performance from it was so inconsistent that we were forced to adapting our lab9 delay function to work at real-time.

Doing this project remotely and across a time-zone difference of 9 hours was challenging. Although we made the best of it, it definitely would have been easier to collaborate in-person.

Getting interrupts to work properly was hard.

**Project Design**

This program is built around a main loop inside of loop() itself. The purpose of this main loop is to allow for easy resetting of the program via break statements. On startup, the system waits for the button to be pushed. Once it’s pushed, an interrupt is used to change some state variables to allow the program to proceed into the “handwashing loop”. If statements within this loop check against conditions that will change program flow such as the user removing their hands prematurely. Subsequent loops handle these events. If the hands are successfully washed, another Boolean variable will allow the fan to turn on along with its subsequent loop. Finally, the system always resets itself at the end of loop().

**-** DESIGN CHOICES –

**Hardware:**

LCD: We opted to use the LCD for all of the timer output as opposed to using a 7-segment. The 7-segment requires a lot of wiring and thus is prone to user error. The LCD is already set up to display text easily so it was trivial to display the time as well.

DC motor (Fan): This seemed like the obvious choice for the fan.

Servo motor (“water faucet”): We opted for the servo motor to simulate the water faucet due to ease-of-use and how the turning of the servo resembles a valve.

Potentiometer: This was necessary to control the LCD screen contrast as the contrast seemed to “drift” between power cycles.

Real-Time Clock (RTC) Module: Although originally intended to control delay timing, this proved too troublesome and instead we downgraded this module to “merely” report the real-time per button press.

DS1307 Temperature sensor: This was the obvious choice for simulating monitoring of water temperature.

Use of analog pins (A0-A15) on the Arduino board: Because of where the positive and negative pins are on the Arduino board, we wanted to use the analog pins which would be closest to the power rail on the breadboard. This allowed us to use shorter jumpers and save the other pins for useful features like PWM and interrupt-handling.

**Software:**

While(system\_enabled): This is the “main” loop of the program. Breaking out of this loop effectively resets the program, which made handling events (such as the user removing their hands) far easier.

myDelay(): As was mentioned with the RTC, inconsistent performance forced us to re-write this function to mirror our Lab 9’s delay function.

ISRs for delay and button pushing: This seemed like a natural choices since these are the only events that truly required interrupts.

Ultrasonic: Rather than tying the proximity of the user’s hands to the relevant while loops, we decided that a separate check made more sense within the loop. This more clearly identifies hand proximity as an unusual source of changes to program flow as opposed to something more predictable and expected like the value of a countdown timer.

Use of state/Boolean variables: Boolean variables are a convenient way to check for certain conditions across the program and they’re relatively easy to debug.