Individual Progress Report

Team #19
John Lee

April 2nd, 2025

ECE 445

TA: Rui Gong

Professor: Michael Oelze

I. Introduction

1.1 Overview

As of April 2nd, the due date of this document, we had several versions of the main PCB with STM32, input and output PCB, trigger board PCB, and multiple water leakage detection PCBs. We have only thoroughly tested the main PCB by connecting other subsystems like the solenoid valve and a pressurized pump; furthermore, the water leakage detection PCBs also work as they should. Within the next few weeks, we aim to solder every PCB, test and verify, and start on CAD encapsulation.

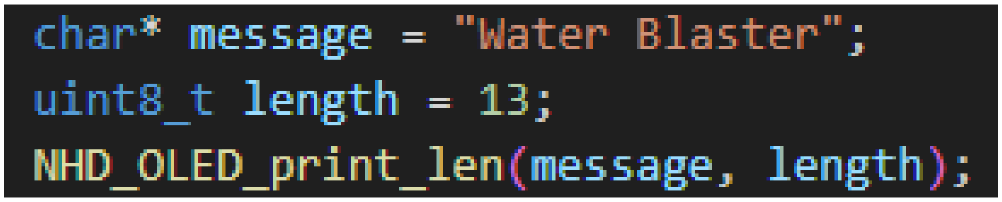
1.2 My Part

I have been responsible for designing, assembling, and soldering different components on the PCB and the firmware used to connect and control the pressurized pump and the solenoid valve. Currently, I am testing PCBs we received a few days ago. In the future, I will be working on the display firmware so that the OLED display shows dynamic status messages.

II. Design

2.1 Firmware

The STM32 serves many different purposes. It has to trigger the pressurized pump and the solenoid valve based on a trigger button. In addition, the microcontroller needs to display useful information, such as the status on the OLED display. Currently, the OLED display only displays a static message that reads “Water Blaster”.



```
char* message = "Water Blaster";
uint8_t length = 13;
NHD_OLED_print_len(message, length);
```

Figure 1: OLED Display

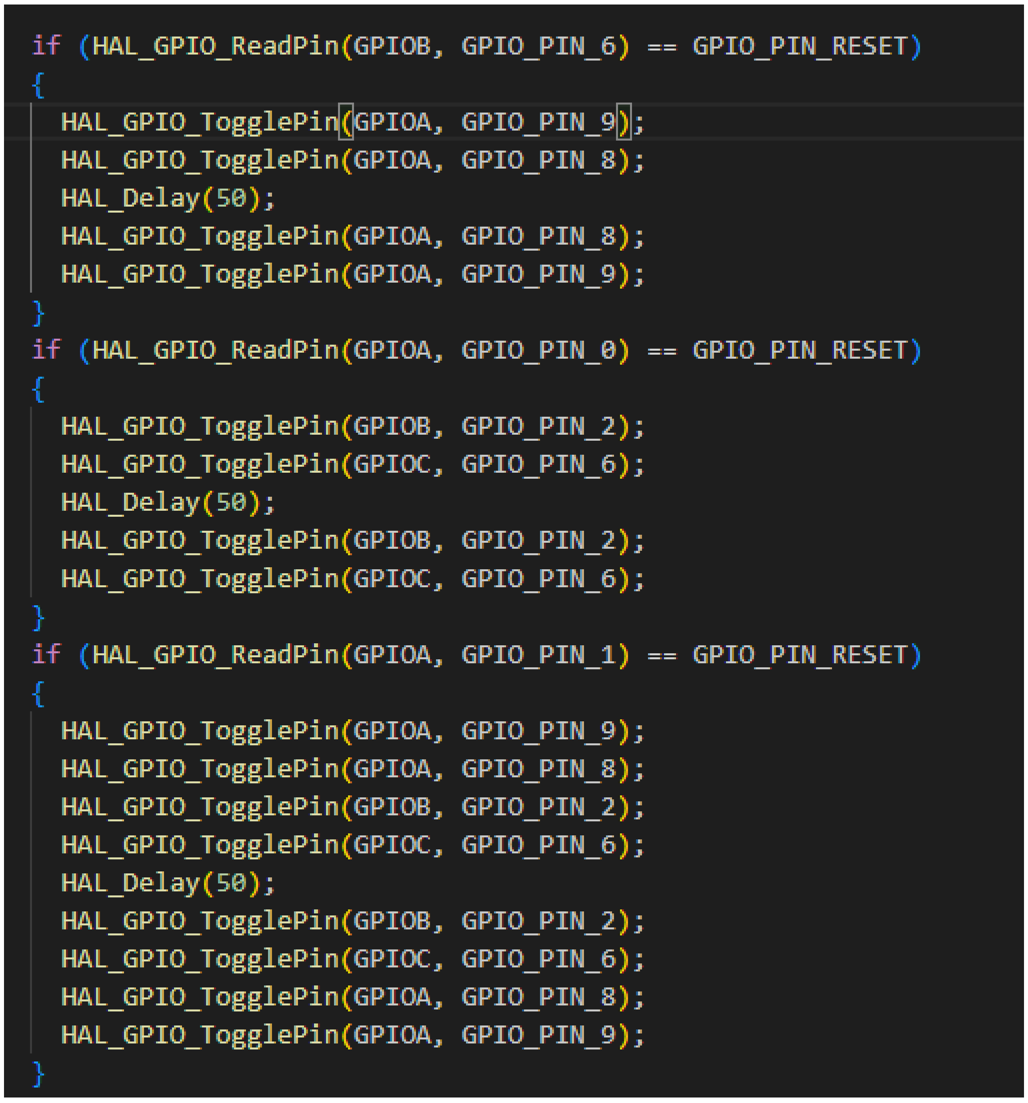
After the initialization of the OLED display, the above code snippet lies inside the main function - it prints the text “Water Blaster” to the OLED display.

Inside the while loop are three different if statements.

1) When button 1 is pressed, which is GPIOB_PIN_6, the solenoid valve is opened by setting GPIOA_PIN_9 to high and GPIOA_PIN_8 to low. After 50 milliseconds of delay, the valve is closed by setting GPIOA_PIN_9 to low and GPIOA_PIN_8 to high.
2) When button 2 is pressed, which is GPIOA_PIN_0, the pressurized pump is activated by toggling GPIOB_PIN_2 and GPIOC_PIN_6. GPIOB_PIN_2 is set to high and GPIOC_PIN_6 to low, simulating the pump's operation.

After a 50-millisecond delay, the pump is turned off by setting GPIOB_PIN_2 to low and GPIOC_PIN_6 to high.

3) GPIOA_PIN_1 (button 3) combines the functionality of GPIOB_PIN_6 and GPIOA_PIN_0. When button 3 is pressed, both the solenoid valve and the pressurized pump operate.



```
if (HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_6) == GPIO_PIN_RESET)
{
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_9);
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_8);
  HAL_Delay(50);
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_8);
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_9);
}
if (HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) == GPIO_PIN_RESET)
{
  HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_2);
  HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_6);
  HAL_Delay(50);
  HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_2);
  HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_6);
}
if (HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_1) == GPIO_PIN_RESET)
{
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_9);
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_8);
  HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_2);
  HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_6);
  HAL_Delay(50);
  HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_2);
  HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_6);
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_8);
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_9);
}
```

Figure 2: Main logic controlling the valve and pump

III. Verification

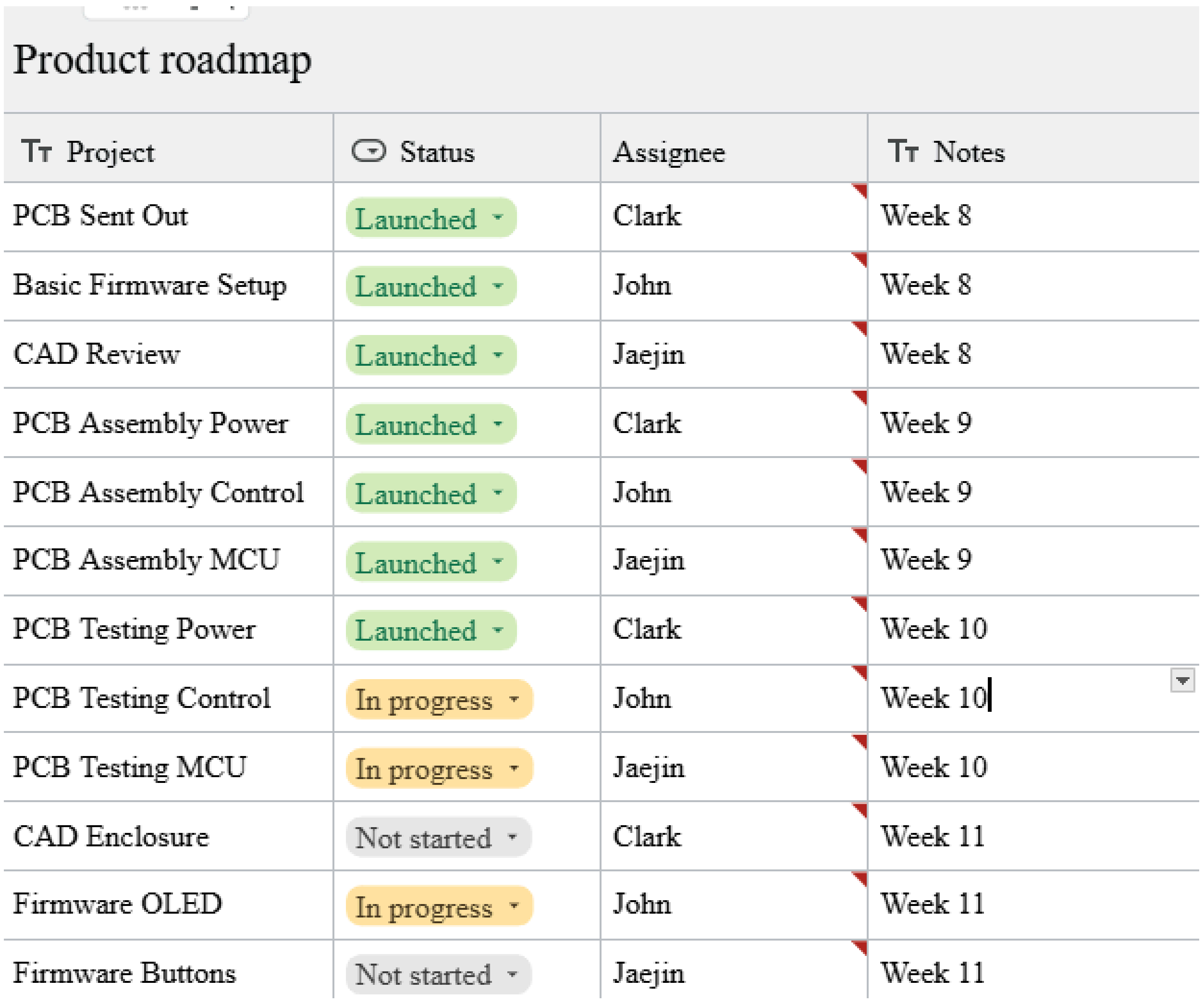
3.1 Test

To test the firmware, we connected the solenoid valve, the pressurized pump, and the OLED display to the main PCB. After uploading the code to the STM32, we connected the PCB with push buttons and verified that the solenoid valve and the pressurized pump function properly. We also saw that the OLED display correctly displayed the message "Water Blaster". In the future, we plan to test out the pressurized pump and the solenoid valve by actually drawing water and firing it out. This will help us gauge one of our three main goals: the range of the electric water blaster. In addition, we will utilize the newly finished trigger board to trigger the pressurized pump and the solenoid valve instead of using push buttons.

IV. Conclusion

4.1 Timeline

I will follow the tentative timeline and schedule that was set on the design documentation.



Product roadmap

Project	Status	Assignee	Notes
PCB Sent Out	Launched	Clark	Week 8
Basic Firmware Setup	Launched	John	Week 8
CAD Review	Launched	Jaejin	Week 8
PCB Assembly Power	Launched	Clark	Week 9
PCB Assembly Control	Launched	John	Week 9
PCB Assembly MCU	Launched	Jaejin	Week 9
PCB Testing Power	Launched	Clark	Week 10
PCB Testing Control	In progress	John	Week 10
PCB Testing MCU	In progress	Jaejin	Week 10
CAD Enclosure	Not started	Clark	Week 11
Firmware OLED	In progress	John	Week 11
Firmware Buttons	Not started	Jaejin	Week 11

Table 1: Tentative Schedule from Design Document

My immediate focus would be to solder the rest of the PCB and test each subsystem. I also aim to extend the OLED firmware to display dynamic status messages.

4.2 Ethical Consideration

We understand the potential risk of the water projectile - the water projectile may be overly strong. After assembling the subsystems, we will downgrade the build if water power is strong enough to cause any harm to anyone, as per the IEEE Code of Ethics and ACM Code of Ethics.

V. Citation

IEEE - IEEE Code of Ethics. (n.d.-a). https://www.ieee.org/about/corporate/governance/p7-8.html

The code affirms an obligation of computing professionals to use their skills for the benefit of society. Code of Ethics. (n.d.). https://www.acm.org/code-of-ethics

UM1725 description of STM32F4 Hal and low-layer drivers. (n.d.). https://www.st.com/resource/en/user_manual/um1725-description-of-stm32f4-hal-and-lowlayer-drivers-stmicroelectronics.pdf