Enhancement One: Software Design & Engineering

Caleb Irwin

CS-499: Milestone 3-2

Professor Gene Bryant

March 23, 2025

Enhancement One: Software Design & Engineering

Briefly describe the artifact. What is it? When was it created?

The artifact that I selected for this enhancement is a smart thermostat program from CS350: Emerging Systems Architectures and Technologies. This artifact was the final project of that course and was created in December 2024. The original artifact utilized a CC3220S LaunchPad, a common IoT development board, to simulate a smart thermostat. The thermostat simulated server communication via UART, increased or decreased the setpoint temperature using buttons, used I2C to sample the current temperature, and simulated heat being turned on by illuminating an LED.

Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development? How was the artifact improved?

I selected this artifact because it demonstrates my ability to develop for embedded systems, design modular software, and integrate networking protocols. This enhancement was particularly meaningful to me because it allowed me to combine my digital electronics background with the software development skills that I've gained throughout the computer science program at SNHU. It showcases my ability to work with both hardware and software. The original artifact utilized UART, I2C, GPIO Buttons & LEDs, and task scheduling to simulate a smart thermostat. My enhancements improved this artifact by implementing Wi-Fi connectivity and two-way TCP socket communication in place of the UART interface for simulating server communication (though I kept UART for debugging, *Figures 1 & 2*). I also developed a Python remote control application with a GUI (*Figure 3*) to allow for easy interaction with the thermostat remotely. Finally, I implemented internet-based time synchronization and daily setpoint scheduling (*Figure 4*).

```
Initializing I2C Driver - I2C Passed
Is this 006? Found
Detected TMP006 I2C address: 41
SPI initialized
SimpleLink started. Connecting to WiFi...
IP Address: 192.168.50.92
GPIO + UART + I2C + WiFi Initialized
Synchronized Time: Sun Mar 30 16:22:42 2025
Tasks Created Successfully

<24,20,0,16:22:44>
<24,20,0,16:22:45>
<24,20,0,16:22:46>
<24,20,0,16:22:47>
<24,20,0,16:22:48>
Client connected!
Sent data: TEMP:24,SETPOINT:20,HEAT:0
Received message: ACK
```

Figure 1 – Initialization Messages Over UART

```
<25,20,0,16:23:11>
Sent data: TEMP:25,SETPOINT:20,HEAT:0
Received message: SETPOINT:26
Setpoint updated to 26
<25,26,1,16:23:12>
Sent data: TEMP:25,SETPOINT:26,HEAT:1
```

Figure 2 – Setpoint Update Messages Over UART displaying Sent/Received TCP Messages

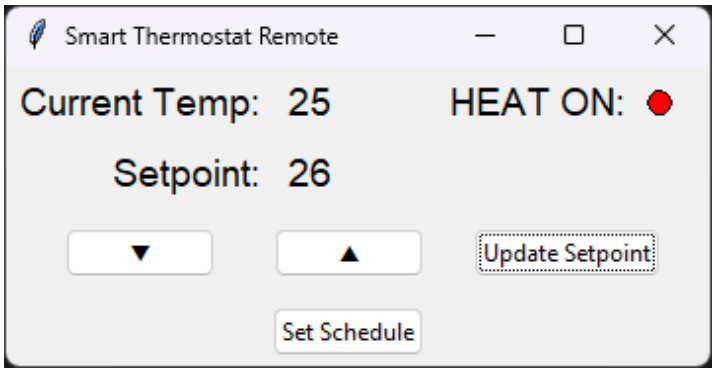

Figure 3 – Python Remote Control Interface, Main View

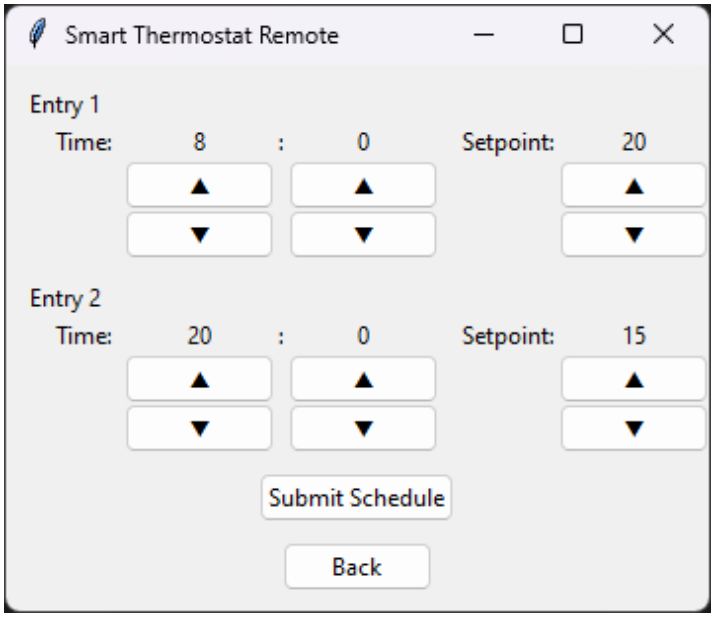

Figure 4 – Python Remote Control Interface, Scheduling View

Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

Yes, I met the course outcomes that I planned to address with this enhancement. Specifically:

[CS499-01] *Employ strategies for building collaborative environments that enable diverse audiences to support organizational decision-making in the field of computer science.*

This project is structured modularly and is well documented throughout the code in order to support collaboration and to encourage future enhancements and cross-functional teamwork.

[CS499-02] *Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts.*

I've documented the system architecture, README file, code, and this narrative with professional quality communication tailored for a technical audience.

[CS499-03] *Design and evaluate computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution while managing the trade-offs involved in design choices.*

This enhancement was designed using algorithmic principles and practices appropriate to embedded systems, including taking into account the trade-offs between real-time performance and design flexibility.

[CS499-04] *Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals.*

In this enhancement I implemented well-founded techniques for embedded systems development, wireless communication, and GUI design that provide value and practical functionality.

I don’t have any updates to my outcome-coverage plan. I still anticipate covering the remaining outcome [CS499-05] in my second and third enhancements, while expanding my coverage of the other outcomes.

Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?

I knew when I planned this enhancement that the targeted improvements were ambitious given the time constraints. My biggest initial challenge was ensuring a properly configured development environment. The CC3220S development board is somewhat dated and requires specific versions of Code Composer Studio, SDKs, System Configuration, and UniFlash to function correctly. I also knew that getting the device connected to Wi-Fi would be the most critical and technically demanding change to the program.

To tackle this, I started with a TI-provided networking example. Even getting the example to build and run took several hours due to configuration challenges. During this phase, I learned a lot about identifying and correcting build issues, flashing SDKs, and working with compiled .sli and .bin files. Prior to this, most of my experience with embedded devices came from working with Arduino microcontrollers and the Dragonfly development board. Through this process, I gained deeper insight into embedded systems development and the complexities involved in integrating hardware with higher-level software applications.

Once I had the networking demo working, I needed to decide whether to maintain the original artifact’s NORTOS implementation or convert it to TI-RTOS. Although the SimpleLink networking stack is usable with NORTOS, I chose to switch to TI-RTOS to take advantage of its real-time

scheduling, multitasking support, and resource management features. This allowed me to run network communication, temperature sensing, and control logic in separate threads, resulting in a more reliable and responsive smart thermostat system.

The enhanced version also incorporates real-time clock functionality and synchronizes with an NTP server once connected to Wi-Fi. This timekeeping feature enables accurate schedule execution for temperature setpoint adjustments throughout the day.

After refactoring the program into TI-RTOS, establishing Wi-Fi connectivity, and enabling NTP synchronization, my next step was to create two-way TCP socket communication between the CC3220S and a Python test program. This test program helped me validate all the required message types that the smart thermostat would send (current temperature, setpoint, heat on/off status, schedule data, and acknowledgments) and receive (setpoint updates, schedule changes, and acknowledgments).

I struggled a bit here, as this was my first time implementing TCP socket communication, but I ultimately got it working as intended. Finally, I developed the Python remote control program. Although I had worked with Tkinter GUIs before, it had been several years, and it took some time to refresh my knowledge and design a program that both functioned correctly and looked the way I envisioned.

Overall, this enhancement turned a simple smart thermostat simulation into a far more advanced, realistic, and extensible system. It has given me confidence in tackling full-stack embedded challenges, from low-level RTOS threading to high-level GUI interaction.