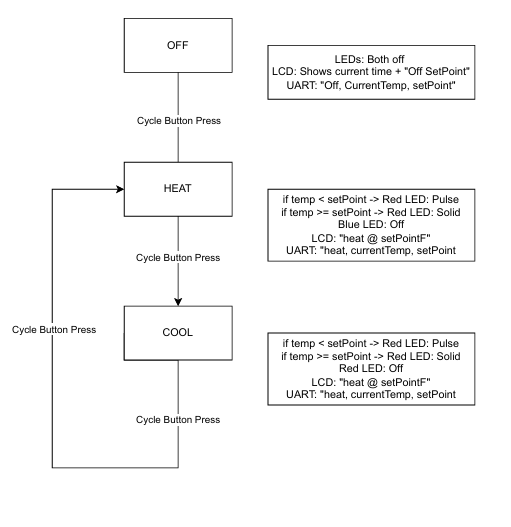
Merrik Wright   
CS350 – Final Project



The development of the smart thermostat prototype provided a hands-on opportunity to integrate software control with physical hardware, emphasizing the principles of embedded systems and state machine design. Beginning with basic GPIO interaction and I2C sensor integration, the project progressively evolved to include UART serial communication, LCD feedback, and responsive LED indicators. Through this process, I gained a deeper understanding of how a microcontroller system interacts with multiple peripherals in real time while managing user input, output display, and data transmission effectively.

The system was designed to cycle between three states: off, heating, and cooling. These states were managed via a finite state machine implemented in Python. A single button toggled the system between states, while two additional buttons allowed the user to adjust the thermostat’s set temperature. Real-time temperature was acquired using an AHT20 temperature and humidity sensor via the I2C protocol, and the current mode and temperature readings were displayed on a 16x2 LCD. Feedback was also provided through red and blue LEDs: when in heating or cooling mode, the respective LED would either remain solid or pulse depending on whether the target temperature had been reached. I had a lot of problems with this. The LEDs would just not work for a while. After a lot of debugging, I was finally able to get it.

A major challenge was managing the hardware-level feedback behavior, particularly with PWM-based LED pulsing. Ensuring compatibility across Python versions and hardware-specific GPIO limitations required several iterations and workarounds. Additionally, synchronizing the LCD updates with temperature reads and UART transmissions every few seconds without creating thread conflicts or timing issues required careful threading and flow control. Despite these challenges, the final result met all functional requirements and performed consistently under test conditions.

Looking ahead to the next phase of development—transitioning the thermostat to a cloud-connected device—requires evaluating potential hardware platforms for production use. Raspberry Pi is a strong contender due to its compatibility with the existing prototype and built-in Wi-Fi. However, Microchip microcontrollers, such as the PIC32 series, offer lower power consumption and are widely used in embedded products. Freescale (now part of NXP Semiconductors) also offers advanced microcontrollers like the Kinetis series with integrated Wi-Fi and extensive peripheral support. Ultimately, the best choice will depend on factors such as cost, power efficiency, and required memory.

Based on this project’s requirements—support for I2C, GPIO, UART, Wi-Fi, and sufficient flash/RAM for the Python logic—the Raspberry Pi remains the most straightforward option for a quick product launch, particularly in low-volume or prototype settings. However, for embedded efficiency, a Microchip or NXP solution with native cloud connectivity and lower power draw may prove more optimal. This project has been instrumental in preparing me to assess these architectural tradeoffs in a practical engineering context. I will be still using my Raspberry pi in the future. I am not sure about the bread board, but the rest I should be using.