Gordon Price

4/18/2025

CS-350

Professor Alexander

**Final Project Reflection**

**Overview**

This project involved building a smart thermostat prototype using a Raspberry Pi, simulating a basic heating and cooling control system with LED indicators. It featured three control buttons, a temperature/humidity sensor, an LCD display, and UART communication to simulate remote data reporting. This hands-on project demonstrated how embedded systems interface with sensors, output devices, and user controls.

**What I Built**

The final working thermostat included:

* **AHT20 Temperature Sensor** for live environmental readings
* **Red & Blue PWM LEDs** to simulate heating and cooling systems
* **3 Push Buttons** to:
  + Toggle between OFF, HEAT, and COOL modes
  + Increase or decrease the target temperature
* **16x2 LCD** to display current time and alternate between temperature and thermostat state
* **Serial UART output** every 30 seconds, reporting system state, current temperature, and setpoint
* A **state machine** to manage all behavioral transitions

**Testing & Functionality**

* Checked buttons using a standalone test script (MultiButtonTest.py)
* Added debug prints to track state changes and temperature settings
* UART output was implemented successfully, transmitting data at 30-second intervals. I verified the functionality using debug statements, even though I didn't run the server simulator script for the final demo.

**Hardware & Wiring**

* **GPIO buttons** for user input (GPIO 24, 25, 12)
* **GPIO-controlled** LEDs with PWM for fading behavior
* LCD interfaced via 6 GPIO pins using digital I/O
* I2C bus used for temperature sensor
* UART **/dev/ttyS0**used for external data reporting

**My Thoughts on future upgrades:**

In a factory environment, I would:

* Replace LEDs with relays or transistor-driven HVAC fans
* Add Wi-Fi and cloud connectivity for remote control
* Include mobile app functionality or web interface
* Use ESP32 or Raspberry Pi Zero W for a leaner and networked setup

**Platform Architecture Comparison**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Raspberry Pi** | **Microchip (PIC32/AVR)** | **NXP/Freescale (Kinetis)** |
| **Capable of Wi-fi?** | *Yes* | *No (Requires External Module)* | *On Some models Yes* |
| **Flash/RAM** | *16GB microSD / 1GB+ RAM* | *32KB–256KB Flash, 2KB–32KB RAM* | *Up to 1MB Flash, 256KB RAM* |
| **Peripheral Support** | *GPIO, I2C, UART, SPI, PWM* | *Strong GPIO, UART, ADC, less I2C* | *Advanced GPIO, I2C, SPI, PWM* |
| **Ease of Development** | *Linux, Python, GPIOZero* | *Needs C / Assembly, MPLAB IDE* | *Typically C but with MCU toolkits* |
| **Best Use Case** | *Rapid prototyping, IoT demos* | *Low-power embedded applications* | *Industrial-grade embedded systems* |