Cheyenne Herder

CS 350

Project

April 21, 2024

The task scheduler created in my project manages when different jobs happen. It decides which jobs are more important and makes sure they get done first. For example, reading temperature is more important than showing a message on a screen.

I designed the scheduler to listen to signals from sensors and timers. Based on these signals, it decides what to do next. It keeps track of tasks that need attention and ensures everything happens smoothly and on time. To visualize how it works, you can think of it like a traffic cop directing cars. I prioritize urgent tasks, like reading temperature data, and handle them first while other tasks wait their turn.

In my thermostat project, I use different components to make everything function. I use I2C to communicate with temperature sensors, UART2 (though I struggled with UART specifically in this project) for device communication, and a Timer to manage time-related tasks.

The project operates on various hardware types, each with unique strengths. One excels in connecting to Wi-Fi for cloud services, another is efficient in data handling, and the third is capable of managing complex operations smoothly.

The thermostat connects to the internet through Wi-Fi, allowing remote control. I ensure data security by using secure methods to transmit data over Wi-Fi, protecting user information.

Regarding memory, the hardware has ample space to store instructions and data for smooth operation. By following coding best practices, I ensure clarity and correctness in the codebase. Overall, this explanation covers how I designed the scheduler, the components used, the different hardware types, Wi-Fi connectivity, and memory usage.