Alfredo Romero

SNHU

CS 350 Emerging system

6/23/2024

7-1 Project Submission

The goal of this assignment was to modify the gpiointerrupt.c file to monitor buttons every 200ms, check temperature every 500ms, and update an LED and report to a server via UART every second. The specific requirements included incrementing or decrementing the temperature set-point by 1 degree upon button presses, turning off the LED if the temperature exceeds the set-point, and turning on the LED if the temperature falls below the set-point. The data sent to the server had to be formatted as <AA,BB,S,CCCC>, where AA is the room temperature in degrees Celsius, BB is the set-point temperature, S is the state of the heater (0 for off, 1 for on), and CCCC is the number of seconds since the board was reset. These tasks were achieved through a task scheduler, a fundamental concept in embedded systems, ensuring that each function runs at its designated interval without interfering with others.

To meet the project requirements, the code was divided into several key sections: button handling, UART initialization and communication, I2C temperature reading, and timer-based task scheduling. The button handling functions, gpioButtonFxn0 and gpioButtonFxn1, were designed to increment and decrement the set-point temperature. These callbacks were triggered by GPIO interrupts configured to respond to button presses. The UART initialization was handled by the initUART function, which set up the UART parameters and ensured successful communication with the server. The initI2C function initialized the I2C driver and configured it to read temperature data from the sensors. Finally, the main thread contained a loop that checked the elapsed time for button presses, temperature readings, and UART communications, updating the LED and sending data to the server accordingly.

However, several issues were encountered during the implementation. Some essential drivers, such as RTOS and UART, did not appear in Code Composer Studio (CCS), leading to compilation errors. This problem required modifications in the uart2.h file, which was outside the scope of the course and posed significant challenges for many students, including myself. Another issue was the program's inability to locate a .out file, a problem that took several hours of research to resolve. These obstacles were primarily due to outdated or incomplete documentation provided by the school, which made troubleshooting more difficult. Despite these challenges, the required code to fulfill the project requirements was written successfully.

The project demonstrated a practical application of IoT principles, where physical devices and sensors communicate over the internet, allowing for remote monitoring and control. The IoT concept envisions integrating sensor data with web-based data and cloud services to create more sophisticated applications than isolated embedded systems can offer (International Conference on Collaboration Technologies and Systems, 2012). The project involved developing an extensible API and a cloud-compatible controller, emphasizing the potential for scalable, high-performance IoT applications. Although the course's documentation issues limited the ability to fully compile and test the project, the implementation of the task scheduler, GPIO interrupt handling, and UART communication provided a solid foundation for understanding embedded systems' operation within an IoT framework.

In summary, this assignment underscored the importance of task scheduling in managing periodic tasks within an embedded system. It highlighted how to handle button interrupts, read temperature data via I2C, and communicate with a server through UART. Despite the encountered issues with drivers and incomplete documentation, the project requirements were met by implementing the necessary code. This exercise not only met the immediate goals of the course but also provided insight into the broader implications of IoT, where interconnected devices can create rich, data-driven applications. Future work could focus on addressing the encountered technical issues and further exploring the integration of cloud-based services with embedded systems to enhance IoT applications' capabilities and performance.

Reference

International Conference on Collaboration Technologies and Systems. (2012). *The Internet of Things (IoT) may be thought of as the availability of physical objects, or devices, on the Internet*. Presented at the 2012 International Conference on Collaboration Technologies and Systems (CTS), 21-25 May 2012, Denver, CO, USA. IEEE. https://doi.org/10.1109/CTS.2012.6261020