Paul Kenaga

Professor Esposito

CS 350

17 February 2023

Project: Technical and Operational Documentation for the Task Scheduler

The purpose of this project was to develop a prototype for a smart thermostat for the company Systec using the CC3220S LAUNCHXL TI development board. The prototype uses the onboard TMP116 digital output sensor to read the ambient room temperature and communicate the value as an integer using the Inter-Integrated Circuit, I2C, serial communication protocol. The value of the current room temperature is compared to a set temperature value. General-purpose input/output, GPIO, pins on the board have been configured to communicate when the set temperature value is greater than the current room temperature by turning an LED on. The LED being turned on simulates a heater being turned on to heat the room temperature to the value of the set temperature. GPIO interrupts were created to allow the set temperature to be increased or decreased by the pressing of buttons located on the sides of the board, LaunchPad User Button SW2 (Left) and LaunchPad User Button SW3 (Right). The Universal asynchronous receiver-transmitter, UART, was implemented to simulate data being transmitted to a server. It outputs the value of the current temperature, the value of the set temperature, if the LED is on or off, and the time, in seconds, since the board has been reset.

The low-level functionality of the smart thermostat prototype is divided into three tasks that utilize the three peripherals, I2C, GPIO, and UART, and execute at three different timer intervals, known as periods. The structure of each task includes a tick function, a value for its period, and a value for the elapsed time since the last call to the task. The first task has a tick function CheckButtons that executes at a period of 200 milliseconds and uses the GPIO peripheral. Global variables left\_button and right\_button are used as input to indicate whether the left button, CONFIG\_GPIO\_BUTTON\_0, or right button, CONFIG\_GPIO\_BUTTON\_1, has been pressed. The left button is configured to the GPIO input pin P04 and the right button is configured to the GPIO input pin P15. Callback functions are used to capture hardware interrupts that occur from either button being pressed. Pressing the left button changes the value of left\_button to TRUE and pressing the right button changes the value of right\_button to TRUE. Switch statements in the CheckButtons function check which button, if any, has been pressed and increments or decrements the value of the set temperature variable, set\_temp, by 1 degree depending on which button was pressed. Pressing the left button decrements the value of set\_temp by 1 degree while pressing the right button increments the value of set\_temp by 1 degree. The values of left\_button and right\_button are reset back to FALSE after each call to CheckButtons.

The second task tick function, CheckTemp, executes at a period of 500 milliseconds and uses the I2C peripheral to capture the ambient room temperature in Celsius. The value is stored in the global variable current\_temp. The function takes the set\_temp variable as input and compares its value to the value of current\_temp. The output of the function is a value written to GPIO pin P64, CONFIG\_GPIO\_LED\_0 , that causes an LED turn on or off simulating a heater. When current\_temp is less than set\_temp the value written to CONFIG\_GPIO\_LED\_0 is 1 which activates the LED. When current\_temp is greater than or equal to the set\_temp the value written to CONFIG\_GPIO\_LED\_0 is 0 which turns the LED off. The global variable heat\_on is another output of the function and used by the UART peripheral to communicate when the heater is on, 1, or off, 0.

The third task tick function, OutputToServer, executes at a period of one second to write data from a memory buffer to the UART interface. The purpose of the function is to report the value of current\_temp, set\_temp, heat\_on, and seconds\_passed to the server every second via the UART peripheral. The variable seconds\_passed represents the time, in seconds, since the board has been reset.

The elapsed time is measured using the timer driver, timer0, and its period is set to 100000 microseconds, which is equal to 100 milliseconds. When the timer’s period has transpired, the timerCallback function is invoked and the global flag, ready\_tasks, is raised. This is useful to keep track of the elapsed time for each task. In the main function of the program, a continuous loop executes and waits for ready\_tasks to be raised before iterating through each task and checking if its period has transpired. When a task’s period has transpired, its tick function is called and its elapsed time is reset to 0. If a task’s period has not transpired during an iteration of the task scheduler then its elapsed time is incremented by 100 milliseconds. After each complete iteration of the tasks, the seconds\_passed variable is incremented by 100 milliseconds, the ready\_tasks flag is reset to 0, and the continuous loop executes again. timer0 operates in a continuous callback mode so that the timer is automatically restarted and will continue to periodically generate interrupts by raising the global flag ready\_tasks.

The project for the thermostat prototype uses a CC3220 device from Texas Instrument, TI, and is part of the SimpleLink™ microcontroller (MCU) platform. It supports the GPIO peripheral through thirty-two programmable input/output pins divided over four ports, the UART peripheral through two UARTS with fully programmable serial interface characteristics, the timer peripheral through four instances of 32-bit user-programmable general purpose timers used to time external events that drive the timer input pins, and the I2C peripheral through an I2C bus that provides bi-directional data transfer over two-wires and interfaces to external I2C devices. The thermostat connects to the cloud via Wi-Fi using a highly-integrated Wi-Fi Network Processor and built-in Wi-Fi Internet-on-a-chip solution with integrated MCU. A CC3220 device has up 256 KB of static random-access memory (SRAM), however, only the CC3220SF comes with an on-chip flash memory of 1024 KB. The flash memory is organized as 2-KB sectors that can be independently erased (CC3220 SimpleLinkTM Wi-Fi® and Internet of Things Technical Reference Manual, 2017).

Comparable hardware architectures to the CC3220 devices from Texas Instrument are PIC32MZ1025W104132 devices manufactured by Microchip and the K32W061 Microcontroller from NXP Semiconductors (formerly Freescale semiconductor). The PIC32MZ1025W104132 devices support the peripherals used in the project by offering sixty-two GPIO pins that monitor and control other devices, three UART modules that feature full-duplex, 8-bit or 9-bit data transmission, and are capable of 2-pin and 4-pin operation with speeds up to 10 Mbps, two I2C modules that offer 7-bit and 10-bit addressing for both host and client operation, and seven 16-bit timers or up to three 32-bit timers. It connects to the cloud via Wi-Fi through a wireless local-area network (WLAN) sub-system that enables TCP/IP based connectivity protocols along with SSL support. The architecture of PIC32MZ1025W104132 devices provides 1 MB of flash program memory, 64KB of boot program flash memory, and 256 KB of SRAM to support the code (PIC32MZ1025W104 MCU and WFI32E01 Module With Wi-Fi® and Hardware-Based Security Accelerator Data Sheet, 2021).

The K32W061 Microcontroller supports the peripherals used in the project by offering two Universal Synchronous/Asynchronous Receiver/Transmitter (USART) interfaces to provide Synchronous and Asynchronous serial communications with external devices, two I2C modules that support direct memory access, twenty-two GPIO pins that can be configured as input or output by software, and two 32-bit timers used to generate periodic interrupts to the CPU. Unlike the other two architectures that offer on-chip Wi-Fi capabilities, the K32W061 Microcontroller would need to be integrated with a wireless solution like Zigbee networking protocol in order to transmit data wirelessly to the cloud using Bluetooth Low Energy (BLE). To support the code, the architecture of the device has 640 KB of on-chip flash memory accessed through a flash controller and 152 KB of SRAM divided into several SRAM instances to increase control of power usage (K32W061/K32W041 IEEE 802.15.4 and Bluetooth LE 5.0 Wireless Microcontroller, 2021).

After analyzing various hardware architectures, my recommendation is to use CC3220SF TI board for development of the next phase of the project. The justification for this recommendation is based on the business requirements that the thermostat must support the peripherals used in the project, connect to the cloud via Wi-Fi, and the architecture chosen must have enough Flash and RAM to support the code. The K32W061 Microcontroller from NXP Semiconductors does not satisfy the requirement of connecting to the cloud via Wi-Fi; it connects to the internet via BLE. The architecture of PIC32MZ1025W104132 devices manufactured by Microchip does efficiently support the peripherals used in the project. Specifically, the temperature sensor must be calibrated to gain good accuracy which creates overhead and increases the likelihood of inaccuracies with temperature readings (PIC32MZ1025W104 MCU and WFI32E01 Module With Wi-Fi® and Hardware-Based Security Accelerator Data Sheet, 2021). The SimpleLink CC3220SF device from TI supports a suite of protocols to connect to the cloud, offers 1 MB of flash program memory, and supports the peripherals used in the project, including a calibrated temperature sensor that offers an Out-of-Box Experience (OOBE). This makes its architecture an appropriate choice for the next phase of the project.

References

*CC3220 SimpleLinkTM Wi-Fi® and Internet of Things Technical Reference Manual*. (2017, February). Texas Instruments. Retrieved February 17, 2023, from https://www.ti.com/lit/ug/swru465/swru465.pdf

*K32W061/K32W041 IEEE 802.15.4 and Bluetooth LE 5.0 wireless microcontroller*. (2021, May). NXP. Retrieved February 17, 2023, from <https://www.nxp.com/docs/en/data-sheet/K32W061.pdf>

*PIC32MZ1025W104 MCU and WFI32E01 Module with Wi-Fi® and Hardware-Based Security Accelerator Data Sheet*. (2021, September 9). Microchip. Retrieved February 17, 2023, from https://ww1.microchip.com/downloads/aemDocuments/documents/WSG/ProductDocuments/DataSheets/PIC32MZ1025W104-MCU-and-WFI32E01-Module-with-Wi-Fi-and-Hardware-based-Security-Accelerator-Data-Sheet-DS70005425.pdf