# CS 350 Final Project

Mark Webster

Dr. Ricky Sethi

SNHU

The TI MCU we used utilized several peripherals including I2C, UART, and GPIO communications, as well as the built-in temperature sensor. The I2C peripheral was used to obtain and read the temperature in into an integer form that we could use elsewhere in the program; the GPIO was used to process button presses to increment or decrement the user-set temperature, and turn the red LED light on based on whether the new user-set temperature was greater than or less than the current temperature we obtained from the I2C. Finally, we used to the UART to serially communicate with our console window on the computer, passing both temperatures and the LED status. For this project, the TI CC3220S was perfect – it packs a punch with 256 KB of flash-based memory and the ability to connect to wifi on any 802.11b/g/n network. Using the OTA library, we can set up the MCU to connect to either HTTP or HTTPS servers using a simple set of four instructions: OTA_init, OTA_set, OTA_get, and OTA_run. These can get used to execute filesets stored on the cloud, allowing automatic updates and seamless filesharing between the local app and the cloud. This would be a killer for allowing our thermostat to be controlled via a mobile app interface.

In terms of competitors for the TI MCU, I also researched two other options: the Freescale MC9S08PB16 and the Microchip dsPIC33CH. The Microchip MCU is a dual-core model made for embedded device with heavy UI interaction. It has a master core and a slave core: the master does the heavy interface lifting while delegating more sub-surface level tasks to the slave. Both cores have ample peripherals to access, including their own UARTs and pulse modulators, and both have dedicated memory—512 kB for the master and 72 kB for the slave. It is a workhorse, except for one small problem—it does not have wifi capability. Therefore, it is best used a stand-alone thermostat.

The Freescale MCU is a decidedly budget option. The smallest of the three in actual size, it still has a wide array of peripherals, supporting serial communication via UART and I2C and two internal clocks for timer usage. However, like the Microchip, it does not support wifi, and it has a low amount of memory at 16 kB. It is made for lightweight applications with very few inputs, and without wifi it simply cannot give us the connectivity we need.

The TI MCU was reasonably priced at $60 and has a huge array of features for that cost. Wireless connectivity is what we need going forward at SysTec, and the TI gives us a great starting point as a prototype.  The Internet of Things is the Internet of the Now, and with great MCUs like this we will be in a pole position to produce top-notch products.

## References

*8-bit 5V Full-featured MCU with Rich Analog | NXP Semiconductors*. (2022). NXP Semiconductors. Retrieved April 18, 2022, from https://www.nxp.com/products/processors-and-microcontrollers/additional-mpu-mcus-architectures/8-bit-s08-mcus/8-bit-5v-full-featured-mcu-with-rich-analog:S08PB

*dsPIC33CH Digital Signal Controllers | Microchip Technology*. (2022). MicroChip Technology. Retrieved April 18, 2022, from https://www.microchip.com/en-us/products/microcontrollers-and-microprocessors/16-bit-mcus/dspic33c-dscs-100-mips/dspic33ch-dual-core-family

T.I. (n.d.). *CC3220R, CC3220S, and CC3220SF SimpleLink<sup>TM* Wi-Fi® Single-Chip wireless MCU solutions</i>. Texas Instruments. Retrieved April 18, 2022, from https://www.ti.com/lit/ds/symlink/cc3220s.pdf?ts=1650785213836