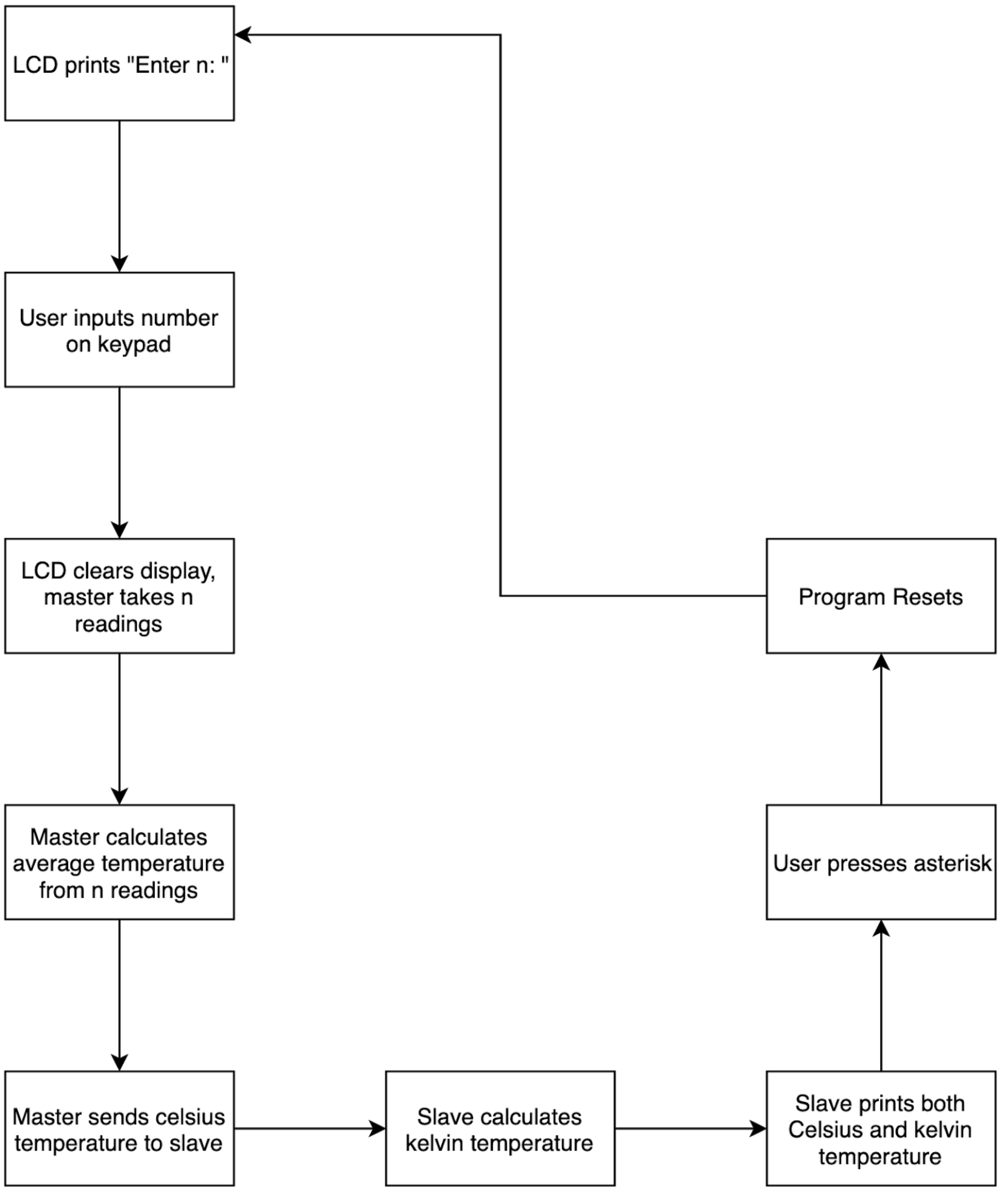
Memo to: Randy Larimer
From: Anthony Louis Rosenblum and Abigail Stroh
Date: March 28th, 2019
Regarding: Lab 4, Temperature Sensor

Summary:


LCD prints "Enter n: "
User inputs number on keypad
LCD clears display, master takes n readings
Master calculates average temperature from n readings
Master sends celsius temperature to slave
Slave calculates kelvin temperature
Slave prints both Celsius and kelvin temperature
User presses asterisk
Program Resets

Amount of memory used by LCD slave:

Flash memory: **1155 bytes**

E000 to E482
FFDC TO FFDD
FFFE TO FFFF

RAM: 68 bytes

Initialization:

LCD Slave:

In order to initialize the S08 the watchdog needed to be disabled, interrupts needed to be enabled, the stack pointer needed to be initialized, and the program code needed to be set at $E000.

Then the I2C registers were initialized to act as a slave within the I2C module
Then certain pins were set as outputs to the LCD and registers were set up for the LCD to display properly.

Keypad Master:

Just like the slave S08 the watchdog was disabled, interrupts were enabled, the stack pointer was initialized, and the program code was set at $E000. The LM19 was attached to the PTB0 pin which was initialized as an input and set up for A/D conversion. The slave address for the LCD was set as 10. The rows of the keypad were set as outputs while the columns were set as inputs. Using the code Daniel provided the I2C registers were initialized so the S08 acted as a master.

Operation:

The LCD slave was cleared and the message “Enter n: ” was printed. The slave would then wait for a numeric input from the keypad and again clear the display when one was received. It would then wait for a hex value from the I2C bus representing a celsius temperature value. Upon receiving that the LCD slave would calculate the kelvin temperature and print “[Celsius Temp]C, [Kelvin Temp]K”. Then the LCD would wait and restart from the top if it received the lookup table value for the asterisk from the keypad.

The master S08 would continue to loop in the mainLoop until a button was pressed on the keypad. When this happened the key_press interrupt (#18) would fire. Using the sum of the row and column hex values, CBEQA commands were initiated and the code would branch to the subroutine corresponding to the button pushed. In each subroutine there was a compare statement to determine whether the high or low slope should be used for the temperature calculation. Once this was done the voltage reading was read in and converted into a celsius value. If a number greater than one was pressed. There was a counter for how many times the voltage was to be read and the program would loop through the subroutine that many times. Adding up all the temperatures as it looped. Then once the counter hit zero the sum would be divided by the number pressed to obtain the average temperature reading. This would be the number that would be sent over I2C in hex form.

Summary Comments:

There were two main problems encountered during this lab. One was getting all of the constants in the temperature equation for the S08 to be in 8bit format. This required using the equation for converting an analog input voltage to a digital value:

ADCR = (Vin *2^resolution/VREFH

In this lab the resolution was 8bits and VREFH was 3.3V. The other main problem encountered was correctly initializing the PTB0 pin on the master. At first we had the 1s and 0s flipped when the APCTL1 register was loaded. This error took a while to find, but after going through line-by-line the error was found and the values flipped.