ECNS-414

Lab #9

November 12th, 2020

Brandon Empie

**Closed Loop Control System in C**

**Objective:** The C programming language provides additional support for mathematical algorithms increasing the capability of embedded systems. The purpose of this lab is to construct and demonstrate a closed loop control system in C using pulse width modulation (PWM). Additionally, a DIP switch will be used to transition from automatic control to manual control via POT1. A temperature sensor LM-335 with a ten millivolt per Kelvin transfer characteristic is to be used. The on-chip Analog to Digital (A/D) converter configured for 10-bit resolution with a reference voltage of zero to five volts will be used to read the current temperature from the lamp.

**Methods:** To accomplish the construction of a closed loop control system a basic timer interrupt was configured to update the lamps condition approximately every 100mS. Within the interrupt, PORTB’s state was checked to determine the control mode. When all DIP switches are off the proportion control algorithm (Output = (32°C – Present\_Value) \* $14 + $10) was implemented to control the temperature of the lamp. If the least significant DIP switch was on the program would operate in manual mode by executing the “1” case statement. While in manual mode the lamps brightness could be adjusted using POT1 as its state is being read with the on-chip A/D converter using channel 0 set for 8-bit operation. The PWM prescaler used for both modes was 128 with a PWMPER3 value of $FA for a period of 8mS.

**Results:** The set point selected was 32 degrees Celsius and the lamps brightness initially went to maximum as predicted. The bulbs brightness decreased as it approached the set point and the filament was just barely light when the set point was reached proving the control system was fully functional. When the DIP switch was flipped (PORTB = 1) the program switched to manual mode operation allowing POT1 to set the lamps brightness as expected. This process was verified by single stepping the CodeWarrior IDE and reading the PWM output using an oscilloscope.

**Conclusion:** Math calculations in C were observed to be significantly more efficient in comparison to assembly. The programmer encountered a unique difference between the two languages regarding data types. In assembly the equation K = 32 – 35 could never result in a value less than zero when using unsigned instructions. The same scenario in C was observed differently. The resulting 8-bit value would reset to the maximum value of the data type when below zero 252 = 32 – 35 causing a runaway condition in the program. This issue was corrected using a standard integer data type to better account for the possible signed/negative solution.