

EET364 Microcontroller Systems

OIT Portland West, Winter 2011

Homework Assignment #7 Due February 24

In this assignment you will write the code (in either Assembler or C) that performs the calculations to solve each problem. You should check your work by running it in the simulator.

A certain OIT instructor works for a company that makes environmental sensors (humidity, temperature, CO₂, CO, pressure). These sensors have voltage (0-5 or 0-10 volt) or current (4-20 mA) outputs. The raw sensor's output is measured by a microcontroller, calculations are performed, and a DAC is set to adjust the output level of the unit.

For this assignment we will use the Dragon12 as a temperature sensor with voltage and current output. The raw sensor is LM45, which is connected to the ADC. We covered that in Lab Assignment 2. The voltage output is the DAC connected via SPI that we covered in Lab Assignment 4. In this assignment we will assume that the ADC has successfully sampled LM45 voltage into word variable *ATD00*, and that it is a 10 bit value such that 1023 is 5 volts and 0 is 0 volts. A word variable *DACVAL* will hold the 10 bit value that will be sent to the DAC such that 1023 is 5 volts and 0 is 0 volts. We are only concerned with reading *ATD00*, doing the calculations and writing to *DACVAL*.

1. In voltage output, 0 volts is 50 °F and 5 volts is 95 °F. What is the code that will read *ADC00*, do the correct calculations, and write to *DACVAL*? Three hints: do all your work in Celsius not in Fahrenheit, you need to scale so that a 5 volt change in output is 45 °F, and you need to subtract an offset after scaling so that 50 °F is 0 volts.
2. An opamp is used to convert the voltage output to a current output. The design of this circuit is beyond the scope of this problem and the course, so just assume it is present. 0 volts out of the DAC will produce 0 mA of current while 5 volts out of the DAC will produce 20 mA of current. We need 4 mA of current to represent 50 °F and 20 mA of current to represent 95 °F. What is the code that will read *ADC00*, do the correct calculations, and write to *DACVAL*?