Fundamentals of Mechatronics (Spring 2021)
ME4405 and BMED4893-A

Lab Assignment Five
Analog to Digital Conversion

Due Friday, March 19th, 2021

Objective: This main objective of this lab is to learn to configure the MSP432 Analog to Digital converter for continuous conversion of a temperature sensor reading. The lab makes use of the DriverLib library, although you are welcome to use direct register access instead if desired.

Deliverables and Grading:

To get credit for this lab assignment you must:

1. Turn in a typed report answering the questions at the end of the lab. This is due **by 11:59pm** on the above due date. You must submit the typed report electronically on Canvas. **(20 points)**
2. Demonstrate proper operation of your code to the TA or instructor. **(30 points)**
3. Submit the commented final version of your code on Canvas. This should be **a single compressed folder** containing your main() function, including header files, etc). **(Pass/Fail)**
4. Take the lab assignment quiz on Canvas, due by **11:59pm** on Friday, March 12th. **(10 points)**

Setup:

This lab requires Code Composer Studio, the MSP432, and the temperature sensor circuit built in Lab 1. The lab uses onboard features of the MCU such as the ADC and UART TimerA. The MSP432 has a user-configurable ADC module known as ADC14. There are 24 such ADC modules which can be used in parallel or individually. Some of these modules have GPIO pins assigned to them to accept analog signals from external hardware.

If you do not have the circuit you built for Lab 1, you can create a new one using new components which are available in the Mechatronics lab. Components for building the circuit are available on the front desk portion of the lab.

Problem Statement:

Write a program that reads the voltage output of the temperature sensor circuit into the MSP via an analog signal pin that corresponds to one of the ADC14 modules. This ADC will convert an analog signal into a digital signal consisting of 10 bits. The ADC module must be configured to scale the input between +Vcc (2.5V) and GND (0V). Care must be taken to tune the sensor output so as to reduce the maximum output voltage to +2.5V. (**Note**: if this is violated, the ADC module may be **permanently** damaged due to high current draw. The ADC is one of the most delicate modules of the MSP.)

Once you are able to measure the temperature from the temperature sensor, you should implement an algorithm to turn on LED2 when the temperature is measured to be above 80°F and turn off when the temperature is below 80°F.

Background:

ADC14:

The MSP432 has 24 Analog-to-Digital conversion modules that can be used in parallel or individually, depending on the application. Each module can be configured to provide 8 bit, 10 bit, 12 bit or 14 bit outputs. Using more bits will provide better resolution (i.e. smaller voltage increments), but also increases conversion time because the MSP432 uses a successive approximation ADC. (More details on the number of clock cycles required for conversion can be found in the datasheet.)

The following functions (as listed in the driverlib reference manual) will be necessary to configure and run the ADC:

- ADC14_enableModule: Enables ADC
- ADC14_setResolution: Sets number of bits
- ADC14_initModule: Sets clock source and clock rate
- ADC14_configureSingleSampleMode: Configures conversion mode
- ADC14_configureConversionMemory: Allocates register to hold results
- ADC14_enableSampleTimer: Configures manual vs automatic triggering
- ADC14_enableConversion: Enables conversion
- ADC14_toggleConversionTrigger: Initiates a single conversion (trigger)

Use the “setResolution” function to configure the ADC to produce a 10 bit output. There are four different modes of conversion; the one used in this lab is Single Channel Single Conversion mode. In this mode, once a conversion is initiated, the ADC will convert the signal into digital data from only one channel and stop without repeating until it is triggered again (by setting the SC bit in ADC14CTL0). (For information on other conversion modes, see the driverlib reference manual.)

Memory should be allocated to save results of the conversion. A function provided in the driver library (ADC14_configureConversionMemory) can be used to configure memory. This function also sets the range of voltages to be used, the channel to be selected and differential vs non-differential (in this lab we will use non-differential, so set this parameter to 0).

After configuration, a conversion can be initiated by triggering the ADC. The ADC_isBusy function can be polled to determine if the conversion is in progress. Once complete, the conversion trigger can be toggled again to initiate a new conversion.

Hardware:

The circuit assembled/built in Lab 1 will be interfaced with the MSP432 in this lab. Adjust the potentiometer which controls the op-amp gain such that the maximum output voltage is 2.5V at 150°F, to be safe. Connect the output of the temperature sensor signal conditioning circuit to the pin that corresponds to the ADC module used. You can decide what ADC module (channel) you would like to use. **Make sure to connect the ground of the Lab-1 circuit to MSP432 ground** so that the entire circuit has a common ground potential.

Software:

The program should convert and display the current temperature at 1 second intervals to the console display in CCS. Configure TimerA such that it produces an interrupt once every second, and place the ADC conversion trigger (and printf statement) inside the interrupt.

After a conversion is complete, print the results to the console display using the printf(...) function. To use printf, you must **include the <stdio.h> header file.** No additional configuration is needed beyond this to use printf, although your first printf statement must start with a newline character (“/n”) in order to begin printing to the console.

When printing, make sure to convert the ADC output to the actual temperature using the appropriate conversion constants.

Furthermore, your program should also turn LED2 red when the measured temperature is above 80°F. When the temperature is not above 80°F LED2 should turn back off.

Requirements:

1. Successfully demonstrate the outcome of the program and all the required functionality to the Instructor or the TA.
2. Submit the commented final version of the code on Canvas.
3. Answer the below questions and submit the typed report on Campus.

Questions:

1. Take data from the above setup for approximately 60 sec. Around 30 sec, heat up the temperature sensor (either by placing your fingers on it or through other means). On the plot, show the two steady-state temperatures achieved by the sensor. How long does it take for the sensor to increase from the first steady-state temperature to the second? (Note: You can copy and paste data from the CCS console window directly to the plotting program of your choice, for instance MATLAB or Excel). **(10 pts)**

2. Calculate the resolution used in this lab in units of mV and in corresponding units of temperature. **(5 pts)**

3. A generator produces a voltage between 0 and 5V, linearly proportional to its angular velocity, which varies between 0 and 100 rad/s. The generator's output voltage is connected to an ADC which is configured with a 14-bit output. Calculate the voltage input from the generator to the ADC and the digital output of the ADC if the angular velocity of the machine is 65 rad/s. **(5 pts)**