Introduction to Mechatronics (ME/AE 6705)

Lab Assignment 5

Analog to Digital Conversion

Objective

The 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. Submit a typed report as a PDF file to Canvas, answering the questions at the end of the lab (2 pages max, can be shorter). This is due at the beginning of class on the due date specified on Canvas. (30 points)

2. Demonstrate proper operation of your code to TAs or instructor during the office or demo hours. You must write your own code for this lab – no lab groups are allowed. (50 points)

3. Submit the commented final version of your code on Canvas (single .c file containing your main.c code, do not submit header files, etc. You should only submit a single file.) (Pass/Fail)

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. 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.

Note: To create a new CCS project for the MSP432P401R target device, and include the DriverLib library in the project, follow the procedure described in Lab Assignment 2. The driverlib folder is available on Canvas.

Problem Statement

Write a program that reads the analog signal given to a pin corresponding 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 (3.3V)* and *GND (0V)*. Care must be taken to tune the

sensor output so as to reduce the maximum output voltage to +3.3V. (Please note if this is violated, the ADC module may be damaged due to high current draw and may be unusable thereafter.)

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 using more bits 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 first step in configuring the ADC is to determine the number of bits to be used. In this lab we will use 10 bits. Secondly, a conversion mode must be determined. ADC14 offers 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`). Other modes available are: Multiple Channel Single Conversion, Single Channel Repeat Conversion, and Multiple Channel Repeat Conversion. More details on these modes can be found in the datasheet.

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).

In total, the functions that must be called to configure and run the ADC are:

- `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)
- `ADC14_isBusy`: Returns a boolean value that tells if a conversion/sample is in progress
- `ADC14_getResult`: Returns the conversion result for a specified memory channel

Please see the documentation for the specific argument list for each function. After complete 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 ADC result can be read using the `ADC14_getResult()` function and 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 used to tune the gain of the op-amp to set the gain such that the output voltage goes only up to 3.3V for 150°F to be safe. Connect the output of the temperature sensor signal conditioning circuit to the pin corresponding to the ADC module used. You can decide what ADC module (channel) you would like to use.

Software:

The program should convert and display the current temperature at 1 sec intervals to the console display in CCS. The ADC should be configured in single channel single conversion mode. The conversion should be triggered at 1 sec intervals using *TimerA*. Configure *TimerA* such that it produces an interrupt every 1 second, and place the ADC conversion trigger (and print 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()`.

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

Requirements

1. Successfully demonstrate the outcome of the program and all the required functionalities to TAs or instructor.

2. Submit the commented final version of the code on Canvas.

3. Answer the below questions and submit a typed report to Canvas.

Questions

1. Take data from the above setup for approximately 60 sec. Around 30 sec, heat up the temperature sensor (either by placing you 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).

2. Calculate the resolution used in this lab in units of mV and in corresponding units of temperature.

3. An ADC is configured with 14 bits output. The output is scaled to 0V to 5V corresponding to the angular velocity of a motor from 0 to 100 rad/s. Calculate the voltage input to the ADC and output of the ADC if the angular velocity of the machine is 65 rad/s.