

## Week 4 Laboratory – ADC

This is the last of the formal laboratories for the course. From now on, you will use the lab sessions to work on the project. The first priority is **Milestone 1 in Week 5**.

### 1. Outcomes:

- To learn how to use the ADC peripheral on the Tiva microcontroller.
- To explore the use of interrupts and ISRs.
- To explore the sampling theorem for analogue-to-digital conversion.
- To explore the use of data buffering in signal acquisition.
- To use a serial link between the Tiva microcontroller and host PC to display status information.
- To make progress towards Milestone 1 for the project (refer to the Milestone 1 specification).

### 2. Source files:

You have been supplied with four new source files for this laboratory: **ADCdemo1.c**, **circBufT.c**, **circBufT.h** and **uartDemo.c**. You will also need **buttons4.h** and **buttons4.c** from last week's lab.

The source files can be found on Learn: **Projects and Laboratories | Laboratory source code | Week-4**

The Milestone 1 specification can also be found on Learn: **Projects and Laboratories | Project Specifications – Milestone 1**.

### 3. Specification for Week 4:

#### 3.1 Set up a new project for the Week-4 Lab (call it “Week-4Lab”).

While this is not strictly necessary, it will help to keep your work in sections. Follow a similar procedure to that you used last week (see CCS 9\_2\_0 Tutorial.pdf and the Week-3 lab sheet, if you need to). Copy the source files from Learn into your new project directory. Remember that you will need to set up the project properties as you did in Week 3 so that links are established to the OrbitOLED and ustdlib modules.

***Note: It is also possible to copy an existing project in such a way that the project properties are maintained.***

Remember that you should only compile the .c files that are part of each program – use right click and “Exclude from Build” as required.

#### 3.2 Compile, link and load ADCdemo1.c (needs access to circBufT.c and circBufT.h)

Run the program and operate the potentiometer on the Orbit board to alter the analogue input voltage between 0 and 3 V. Observe the 12-bit integer that results. Study the source code for ADCdemo1 in detail to understand how it works.

3.3 **Prepare a new version of the program** (say, ADCdemo2.c or miles1\_v1.c) to change to the analogue input AIN9, PE4 (J1-05). Set up the bench power supply to provide a voltage between 0 and +3 V and set the current limit to 0.01 A. Use two banana-to-alligator cables and two of the mini cables that came with your Tiva to connect the input voltage and ground to the board.

**Take care to check the voltage & polarity of the applied voltage BEFORE you turn on the supply output.**

Check the operation of the program. The relationship between the displayed value and the voltage should be perfectly linear.

3.4 **Work out** a suitable size for the circular buffer for two different applications:


3.4.1 For reading the position of the potentiometer if it is to be used as an input to the fitness monitor UI.

3.4.2 For sampling data from the accelerometer, which will respond to the wearer's footsteps. Also work out suitable sampling rates for both peripherals (refer to the Milestone 1 and 2 specifications for the project). Present a case in your lab book for the values you determine. It is reasonable to assume that the motion of a person running is limited to < 3 Hz. Test your sampling/buffering/averaging system by means of the function generator.

**Again, check the voltage of the function generator output with the scope BEFORE you connect to the ADC input.**

3.5 Use code from **uartDemo.c** to send the same data that is on the OLED display to the Tiva board's host PC. Rather than sending every ADC reading, data should only be sent at predefined intervals of about 2 Hz. To view the serial data in CCS you will need to create a new terminal connection:

3.5.1 First, open the Terminal pane by clicking View > Terminal. (You may prefer to use a separate application, TeraTerm for this.)

3.5.2 With the Terminal pane open, click the "Open a Terminal" button , or Ctrl+Alt+Shift+T.

3.5.3 In the window that opens choose Serial Terminal, with 9600 baud, 8 data bits, 0 parity bits, 1 stop bit, no flow control and 5 seconds' timeout.

3.5.4 The Port will vary from USB port to USB port and PC to PC. Open the Windows Control Panel, select Device Manager and expand the Ports (COM & LPT) group. The name in brackets next to the entry Stellaris Virtual Serial Port is the name that should be used for the serial terminal port.

NB: The topic of serial communication will be covered in lectures in Week 7.

3.6 **Carefully read the Milestone spec** and make sure you understand exactly what is required.

Carry on developing your code. Remember that the compliance of your program for Milestone 1 will be checked during your lab session in Week 5. **Don't leave your preparation to the last minute.**

#### 4. **Guidance:**

- Prepare the new programs one step-at-a-time.
- Make new code as modular as possible, using appropriately designed and named functions. This will pay off when you come to build a much bigger program to drive your fitness monitor.

- Test, test, and test once more. Make sure your tests cover the specification fully. A lot of problems with software occur because the developer has not been thorough in testing their code. Record how you have tested your code and save any special programs you may have used for testing.

----:----