CENG 412 Embedded Systems
Lab 4 A to D Converter

Name ___________________ **Signed** _______________

Introduction
In this lab you will use the A/D converter of the HCS12 to measure a DC voltage. You will write an Assembly language file and call this program from a C code file.

Equipment
1. HCS12 workstation board with Demo board attached.

Procedure - Part A
1. A variable DC voltage will be derived from the potentiometer circuit on the **Demo** board. The potentiometer is a round thumbwheel next to the LED display. Consult the Schematic diagram for the **Demo Board** on the course web page and locate R12, the 10 kΩ potentiometer. Trace the wiper connection of the pot back to the P1 Primary I/O connector.

   What is the pin number of the potentiometer wiper on the P1 header? ___________

2. Examine the HCS12 Workstation Pinouts link on the course web page and verify the pin number for the pot and verify that the pin number found in step 1 is connected to channel 2 of the A/D converter**.**

3. Connect a DC voltmeter to this pin and verify that the wiper voltage varies from 0 to 5 volts DC. Set the potentiometer voltage to 2.50 volts.

4. Create a new project in CodeWarrior with a C program and an Assembler program. The C program will be the same as in the previous lab. Refer to the sample A/D program file shown at the end of this lab. Modify the code as follows:

   Find and enter the values for all the missing addresses for the A/D enable register, the A/D control registers, the A/D Status register and the A/D result registers.

   Determine the values needed for the admask2 (ATD0CTL2), admask3 (ATD0CTL3), admask4 (ATD0CTL4) and admask5 (ATDOCTL5) parameters to configure the A/D converter as follows

   - Power-Up the A/D converter
   - 4 conversions
   - 8 bit converter SMP0 & SMP1=0, PRS4 - PRS0 – set the pre scale value to 00101
   - unsigned, right justified results, single scan on channel 2 only

   Enter the values for admask2, admask3, admask4, and admask5 in the program code.

5. **Make** the project in CodeWarrior**.** Open the **Project.map** file and locate the **Entry point** address of your code in the STARTUP section. This will be close to the beginning of the file. Scroll down to almost the end of the Project.map file (just above the UNUSED-OBJECTS SECTION) and locate the address of the label **results**.

   Entry point address __________ results label address __________

6. Open AsmIDE and download the s19 file**.** Run the program - g <entry point address> - and locate for the results at the address you found above - md <results label address>. Record all 4 values. Calculate the expected value for the conversions - 2.5 volts is the point value between 0 and 5 volts. What is the midpoint value between 0 and the full scale value FF in hex? Mid-point value = ____________

7. Run the program again with the voltage set to 1.0 volt and then 4.0 V. **Record all these values in a table**. Verify they are correct. **Each value is 4 hex digits – 2 bytes – long.**

Analog voltage	**Measured digital value – 4 values**				**Calc digital value**
1.0 V					
2.5 V					
4.0 V					

Modify the program to do the following

- 10 bit converter with 4 conversions
- signed, left justified results, single scan on channel 2 only

Record the values in the table.

Analog voltage	**Measured digital value – 4 values**				**Calc digital value**
1.0 V					
2.5 V					
4.0 V					

Procedure - Part B Continuous Conversions

1. Create a new project in CodeWarrior. Open the main.asm file from Part A and copy/paste the assembler code from this project to your new main.asm file. Edit this code to make the red LED display on the Demo board show the value of the A/D conversion continuously. You will need to change the following:

   - modify the **admask3** parameter for 1 conversion
   - modify **admask4** for an 8 bit conversion
   - modify **admask5** for right justified, unsigned results, continuous conversion sequences and sample only channel 2

2. You will need to define the address for PORTH and DDRH. configure Port H as an Output port and transfer the contents of the Result Register to register D and then send this value to Port H. Modify the last part of code as shown below.

```
Again:  brclr   ATD0STAT0,SCFflag,*   ;wait for conversion to complete
        ldd     ATD0DR0               ;get result
       stab    PORTH                  ;send to port H
       bra     Again
        rts
```

3. Vary the R 12 potentiometer and check to see if the display varies as well. Measure and record V_{in} for 1 bit only high (0000 0001) and the lower 7 of 8 bits high (0111 1111).

V_{in} (0000 0001) ________________ V_{in} (0111 1111)   ___________________

Demonstrate the final working version of all your programs.

```
;*******************************************
;*     A/D Converter
;*       Name:
;*       Date:
;*       Course: Embedded Systems
;*     uses ATD0 converter, 8 bit conversion, 4 conversions, unsigned
;*     right justified results, Single scan, channel 2
;*
;*********************************************

; export symbols
         XDEF asm_main

;register addresses

ATD0DIEN:     equu              ;A/D enable register address
ATD0CTL2:     equ              ;A/D Control Register 2 address
ATD0CTL3:     equ              ;A/D Control Register 3 address
ATD0CTL4:     equ              ;A/D Control Register 4 address
ATD0CTL5:     equ              ;A/D Control Register 5 address
ATD0STAT0:    equ              ;A/D status register address
ATD0DR0:      equ              ;Result register 1 address
ATD0DR1:      equ              ;Result register 2 address
ATD0DR2:      equ              ;Result register 3 address
ATD0DR3:      equ              ;Result register 4 address
ATD0DR4:      equ              ;Result register 5 address
ATD0DR5:      equ              ;Result register 6 address
ATD0DR6:      equ              ;Result register 7 address
ATD0DR7:      equ              ;Result register 8 address

;values of parameters

Delay:         equ    $600     ;Delay value
SCFflag:       equ     $80     ;detect conversion done byte
admask2:       equ             ;ADPU = 1 Enable A/D
admask3:       equ             ;do 4 conversions
admask4:       equ             ;8 bit conver,2 A/D conver cycles, prescale 00101
admask5:       equ             ;Right justified, unsigned, single scan, channel 2

;memory storage
results:       ds.w    04

; Code section - this assembly routine is called by the C application
asm_main:
      movb      #$00,ATD0DIEN           ;disable digital inputs
      movb      #admask2,ATD0CTL2       ;configure ATD0CT12
      ldx       #Delay                  ;Delay 100 usec - A/D powers up
loop: dex
      bne       loop
      movb      #admask3,ATD0CTL3       ;configure ATD0CT13
      movb        #admask4,ATD0CTL4     ;configure ATD0CT14
      movb        #admask5,ATD0CTL5     ;start conversion
      brclr       ATD0STAT0,SCFflag,*   ;wait for conversion to complete

      ldd         ATD0DR0       ;get 1st result
      std         results       ;store in memory
      ldd         ATD0DR1       ;get 2nd result
      std         results+2     ;store in memory
      ldd         ATD0DR2      ;get 3rd result
      std         results+4    ;store in memory
      ldd         ATD0DR3      ;get 4th result
      std         results+6     ;store in memory
      rts                      ;return to caller
```