EE 335 Advanced Microcontroller Engineering
Oregon Tech Portland, Winter 2014

Lab Assignment #2 — Analog to Digital Converter
Due January 23

Objective:
The student will demonstrate the usage of the Analog to Digital Converter in the 68HCS12.

Equipment and Software needed:
- Everything you needed for Lab Assignment #1
- A signal generator and an oscilloscope (available in the classroom)

General Instructions:
Note that Freescale Semiconductor refers to an Analog to Digital Converter as an “ATD” rather than the conventional “ADC.” There are two ADCs in the microcontroller. We will restrict ourselves to the first ADC. The Dragon12 board connects most of the pins to various devices on-board. Examine the documentation to find an unused input for this assignment. An input voltage of 5 volts will generate ADC value 1023 while an input voltage of 0 volts will generate an ADC value of 0.

The Dragon12-Plus board provides a potentiometer giving an adjustable voltage input of 0 to 5 volts on PAD07 that can be used for static voltage measurement. You can do initial program debugging with the static input or in the simulator (which also has a static input). You can also use the square wave signal generator that is on the Dragon12-Plus board. However to complete the assignment you will need to connect a signal generator to an ADC input. It is recommended that you prepare for this final operation in advance of the class meeting where you can use the equipment in the class.

WARNING! WARNING! You must be careful that the voltage not go below zero or above 5 volts, otherwise you could damage the 68HCS12 ADC input, converter, or the entire microcontroller. This happened to a couple of students — the board is expensive so don’t let it happen to you. **Check the voltage with the oscilloscope before connecting to the board.** The signal generator screen displays the voltage when the output is connected to a 50 ohm load. Since the 68HCS12 has a high impedance input, the voltage will be twice that indicated on the signal generator display. You will also need to set an offset voltage otherwise the sine wave output voltage will be centered around zero volts.

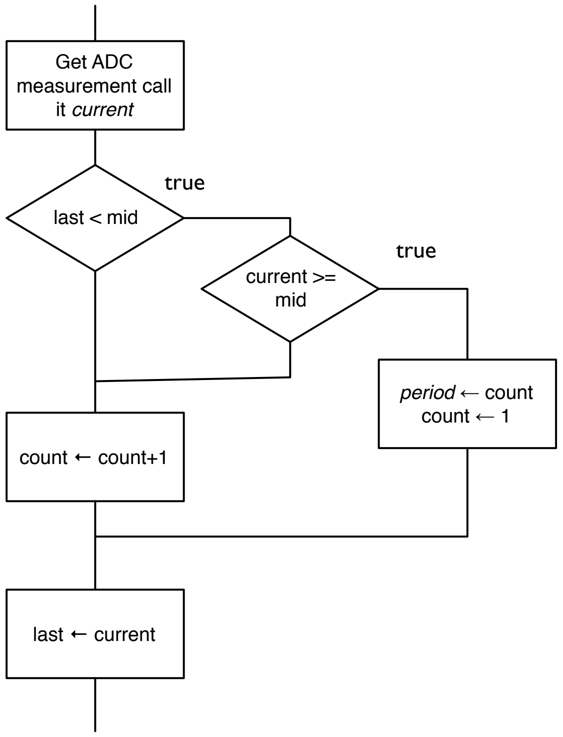
Program Design:
In this unit, a waveform will be examined. The ADC must be configured with the SCAN bit set for continuous operation, 8 conversions, and for 10 bit conversion.Set the AFFC bit so that reading a result register will clear the SCF flag, allowing the next interrupt to occur. With this configuration, the ADC will run continuously,

converting at the maximum rate, with an interrupt occurring every 8 conversions. The eight conversion results will need to be processed in sequence (use iteration for this) in the interrupt routine before returning.

You will need to measure the minimum, maximum, and mean voltages as well as the frequency of the input signal.

To measure the minimum and maximum voltages, define two 16-bit variables, say *min* and *max*. In your program code, initialize min to 65535 and max to 0. In the interrupt routine examine every measurement value and if the value is greater than the current value for min, replace min with the measurement value. If the value is less than the current value for max, replace max with the measurement value. There are instructions (which we didn't discuss in class) that make this task easier — look up the instructions *EMINM* and *EMAXM* in the textbook.

Measuring the period requires measuring the time between rising mid-point crossings, which is a somewhat more involved task. Calculate the mid-point to be $mid = \frac{\max + \min}{2}$.The flowchart on the right describes an algorithm that when executed for each measurement value (in sequence) will calculate the period in units of ADC measurement time. It uses variables named *period, count,* and *last*. *Current* is the measurement value currently being examined.



To calculate the mean voltage requires averaging the voltage over time and can be done using the same low-pass filter technique used in EE333 Fall 2013. Create a 32-bit variable *mean* that will contain the mean voltage times 65536. By picking a value that is a power of 2 and the size of a word, the calculations become easier. Each measurement value gets added into the variable using the assignment mean = mean*65535/65536 + measurement. However mean*65535/65536 is the same value as mean - mean/65536. We can "calculate" mean/65536 by just using the upper 16-bit word of mean. The additions and subtractions will have to be carried out using 8-bit add and subtract and add with carry and subtract with borrow instructions since there are no 32 bit addition instructions in the 68HCS12.

The measurement data should be examined at the start of the interrupt routine because there is little time between the interrupt and the value in ADR00 being overwritten with a new sample. For that reason it should also be examined in order — don't do the minimum. maximum, and period calculations in three separate iteration loops, but all in a single loop so the the value in ATD0DR0 can be fetched once and early before it gets overwritten with the next value. Then after doing the min, max, and period calculation with the value in ATD0DR0, do it with the value in ATD0DR1 and so on through all 8 ADC values.

The idle loop (idle process) should convert the value in *min* to a voltage in millivolts and store in a variable *minvolts*, convert the value in *max* to a voltage in millivolts and store in a variable *maxvolts*, the value in *mean* to a voltage in millivolts and stored in a variable *meanvolts*, and finally convert the value in *period* to the frequency in Hertz and store in a variable *frequency.* The frequency in Hertz is 1,000,000 divided by the period in microseconds. It then repeats the process so that *minvolts, maxvolts,* meanvolts*,* and *frequency* are the recent calculations. These calculations should be done in the idle loop to minimize the time in the interrupt service routine.

Testing the Program:
Set up a function generator to generate a sine wave, 4.5 volts peak-to-peak with a 2.5 volt DC offset. Set the frequency to 100 Hz. Run the program a few seconds, then abort and use the MD command to get the values *minvolts, maxvolts, meanvolts*, and *frequency*. Compare to the expected values, which should be 250 millivolts, 4750 millivolts, 2500 millivolts, and 100 Hertz, respectively.

Don't forget to include the heartbeat code! The heartbeat light should continue to flash until the Abort button is pressed.

To turn in:
- Commented program listing
- Values of *minvolts*, *maxvolts*, *meanvolts*, and *frequency* you observed. Discuss your results.