Name _______________________________

Partner (s) ___________________________ Grade _______/10

Introduction
The goal of this lab is to learn to apply proper sampling theory when gathering data using an Analog-to-Digital Converter (ADC) system.

Objectives
• Learn about using A/D converters;
• Learn proper technique for sampling an analog signal;
• Learn how to interface an ADC board to Matlab;
• Learn the basics of signal processing.

Equipment Provided
• Oscilloscope;
• Signal generator;
• Computer with ADC and connection boards (new).

References
• Instrumentation text book;
• Course web page.

Procedure
Your lab this week includes a new piece of equipment – the Microstar DAP840 analog-to-digital (ADC) interface board. The board is installed internal to the computer at your work station. You connect signals to the A/D board via a ribbon cable and connector strip shown at the right. The first thing to do for this lab is to setup the Agilent signal generator to output a signal which you will capture via the DAP840 board and read into Matlab using a script that has been written for you.

1) Connect the Agilent 33250A signal generator output to the DAP840 board by connecting the signal generator ground to the G0 terminal of the A/D interface board and the signal generator output to the S0 terminal of the A/D interface board. Use the little white “pry bar” to open the connection port. You can see the pry bar on the long grey strip at the right. Take care that you do not lose this bar.
2) Set the signal generator to output a sine wave with an amplitude of 2 volts peak-to-peak, 1-volt DC offset, and a frequency of 500Hz.

3) Use the Tektronix scope to verify your waveform looks like it should at the connection terminals to the A/D board. This is a good time to learn more about the features of the oscilloscope, so put the scope in **DC Coupling** mode and then use the Measure function to find the peak-to-peak voltage, the frequency, and the RMS voltage and record the values below. Calculate what the RMS voltage should be (research this on the web if needed) and record below.

\[ V_{pp} \quad \text{V} \]
\[ \text{Freq} \quad \text{Hz} \]
\[ V_{RMS,meas} \quad \text{V} \quad V_{RMS,calc} \quad \text{V} \]

4) Place the scope in **AC Coupling** mode, which removes the DC component of the signal it is displaying. Record the RMS voltage from the scope and the calculated RMS value.

\[ V_{RMS,meas} \quad \text{V} \quad V_{RMS,calc} \quad \text{V} \]

5) What conclusions can you draw about the method used by the scope to calculate RMS values?

6) Place your scope back in the **DC Coupling** mode. Start Matlab by selecting the icon from the Windows 10 applications tray (wait for it, these computers are really slow)!
7) Use the msgets0 function in Matlab to sample the waveform. The format of the function is:

\[
\text{data, time} = \text{msgets0(rate, numpoints, gain)}; \quad \% \text{Reads A/D channel 0 “single-ended”}
\]

\[
data = \text{vector of voltage values returned from the converter, in decimal};
\]
\[
time = \text{time vector corresponding to the data points};
\]
\[
rate = \text{number of samples per second};
\]
\[
numpoints = \text{number of data points};
\]
\[
gain = \text{optional selectable gain (1, 10, or 100 with a default of 1)};
\]

Here is a sample program to acquire and plot data:

\[
[d, t] = \text{msgets0}(2000, 100); \quad \% \text{rate=2000 samples per second, numpoints=100, gain=1}
\]
\[
d = d * 5 / 32768; \quad \% \text{Convert A/D counts to volts – the DAP840 returns the}
\]
\[
\quad \% \text{signed 14-bit results in a 16-bit word with zeros padded in}
\]
\[
\quad \% \text{the two LSB positions.}
\]
\[
\text{plot(t, d)}; \quad \% \text{Plot the waveform}
\]

8) **NOTE:** There has been some strange incompatibilities between the DAP840 board, Windows 10, and Matlab, especially as new software versions are installed. It appears that the problems can be avoided by executing your Matlab code from a .m file rather than the Matlab editor window, although on the day I last tested this script it ran fine from the editor window. Alternatively, executing from the command window *usually* works. For today, I recommend downloading the lab32019.m file from the class web site and saving it to your N:drive. Then, execute the Matlab script. A copy of the file is shown below:

```matlab
close all;
clear all;

Sample_frequency = 50000; \quad \% \text{sampling frequency}
number_points_to_gather = 10000;

[d, t] = msgets0(Sample_frequency, number_points_to_gather);
\% Convert ADC values to voltages
\% KM = 500;
numberCycle = 2; \quad \% \text{desired number of waveforms displayed}
ptswave = Sample_frequency / knownF; \quad \% \text{number of data points per wave}
\% length = numberCycle * ptswave * (1 / Sample_frequency);
length = numberCycle * ptswave * (1 / knownF);

plot(t, d);
axis([0 length -1.1 1.1]);
hold on
plot(t, d,'ok');
axis([0 length -1.1 1.1]);
msstop; \quad \% \text{This needs to close the execution otherwise frequency values don’t seem to work.}
```

9) Modify the .m file, if necessary, to display *approximately two periods* of the waveform on a Matlab plot. Run the program and print your plot. Also, print a copy of your code.
10) Modify the script as indicated below and comment on your results. Capture 500 data points. Set the DC offset to 0V.

<table>
<thead>
<tr>
<th>Sample Rate (Hz)</th>
<th>Frequency(Hz) (Agilent)</th>
<th>Amplitude (V_p) (Agilent)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>20,000</td>
<td>500</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2,000</td>
<td>500</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>500</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

11) Using the **Arb** function on the signal generator, choose **Select Waveform**, then **Built-in waveforms**, then the **Cardiac** option to output an approximation for an EKG signal. Record the lowest frequency that you can sample at and still see the entire waveform. Print a plot.

   Sampling frequency __________ Hz

Now that you understand the basics of signal acquisition using an ADC board and Matlab, the next step is to do some signal processing from within Matlab.

12) Set your function generator to output a sine wave of 1000Hz, 0.0 V offset, and 2.0 V_{pp}.

13) Capture the waveform in Matlab. Then, add Matlab code to the lab32019.m function to calculate and display the following parameters on a plot of the waveform. Note that there may be built-in Matlab functions to do some of these. Print the plot.
   a. Frequency
   b. Period
   c. Peak-to-peak voltage
   d. RMS voltage (figure out how to do it)

14) Repeat step 13 for a square wave.

15) Repeat steps 12 and 13 but add a 1.0 V offset to the sine wave.

16) Run the Matlab function **msstop** to shut down the connection between A/D board and Matlab. Then exit Matlab before signing off.

**To Turn In**
- This handout along with your plots.