**Digital to Analog Conversion**

In this lab, we will generate an analog signal using a digital-to-analog converter. The frequency of the input signal will be chosen by you, but should be fairly high so that it will display well on the lab oscilloscopes.

The waveform will be a sum of sine waves: $x(t) = 1.024 \times sin(2 \times \pi \times f \times t) + sin(5 \times \pi \times f \times t)$

Please note that this has maximum amplitude 2.048 Volts.

To make the waveform smooth, you need to generate a new sample as fast as possible. One of the things you should include in your lab report is an analysis of the time required to generate your samples and how that affects the maximum sample rate. It would be nice if you could generate a good wave for $f$ in the range 100-10000 Hz, with a sample rate in the 50-100KHz range.

The DAC's that we use are included in your lab kit. They are little 8-pin devices (except they're the MCP4821 rather than the one I specified). The MCP4821 differs from the MCP4921 in that it has an internal voltage reference. You interface to them via the SPI (Serial Peripheral Interface) interface (that's triply redundant, I know).

This will require you to become minimally familiar with the MSSP device, although we'll write code in discussion that will make this easy, and post it on-line. If you want, you can write your own.

This will require you to use two pins (the SDO and SCK of the SPI interface – the DAC is an SPI receiver only, so there's no input on SDI) plus a third pin, the DAC $\overline{CS}$. The way you'll send data to the DAC is to drive the $\overline{CS}$ low (active), then send 16 bits of data (two bytes), then drive the $\overline{CS}$ back high.

The format of the data can be found in the DAC datasheet, but, for all those who don't really love reading datasheets, here's the story: the way the DAC is interfaced, you can just write a bit pattern to it. To make it work right, send a 16-bit value whose most-significant 4 bits are 0001, and whose least-significant 12 bits are the 12-bit encoding of the voltage you wish to see on the output, with 0 producing 0V and 0xfff producing about 2V (you'll recognize this as a "scaled" representation, I'm sure).

Here's the drill in bulletized form:

- Initialize the MSSP for SPI operation.
- To send a new value to the DAC, drive the chip select to 0, then send

your data, then set chip select high. You'll have to send 16 bits per sample.

- Start with a sawtooth wave with a 100 Hz frequency (so you can debug this part first).
- Your program should have a main loop which is empty. All work should be done via the ISR.

To get the desired accuracy in the time domain you will need to use one of the timers to provide a time base.

To get credit for this exercise, demonstrate your working model for the TA.