Homework Assignment #4

EE475 Fall 2003

Assigned: Thursday, October 30, 2003
Due AT THE START OF CLASS on Thursday, November 13, 2003

A simple signal generator can be constructed by sending a periodic sequence of digital values to a digital-to-analog converter. The periodic sequence can either be calculated "on the fly" or a stored data table can be used.

If you want to generate a periodic signal with frequency *f*, the mathematical expression is

$$A \cdot \sin\left(\frac{2\pi f n}{f_s}\right), \quad n = 0,1,2,\cdots$$

where *A* is the desired output amplitude and f_s is the *sampling rate* of the digital-to-analog converter (DAC). If you are able to use floating point math, the expression above can be implemented directly in software. However, it may happen that floating point math is too slow, or it may require too much program memory if the processor does not have floating point hardware. In this case it may be possible to pre-calculate one cycle of the desired sinusoid in advance and store it in a static memory array. You will use both techniques in this homework assignment. Rather than writing to a real DAC, you will write the results into a file as a sequence of integer (text) values.

Problem #1: Write a C program that generates several cycles of a sinusoidal waveform using the floating point formula above, and write the result as a text file. The output needs to be 16-bit 2's complement integer numbers in text form, so round the floating point number to the nearest integer before writing to the file. Use *A*=32767, and f_s = 5 kHz in your code. The command line for your program should be something like:

```
sine_gen  f  cycles  out_file
```

where `sine_gen` is the name of your program, `f` is the desired sinusoid frequency in Hz, `cycles` is the integer number of waveform cycles to produce, and `out_file` is the name of the output text file to be created.

Try several different frequencies and observe several output cycles: comment on the periodicity of the output data. What happens if the frequency *f* is not an integer factor of 5 kHz?

Problem #2: Now modify your C program or create a new program that uses a sinusoidal lookup table (as discussed in class) to generate the output sequence *instead* of calculating the sin() function on the fly. You must use the include file "`sinetable.h`" from the course web site to define the lookup table data.

Use the same command line format as for Problem #1.

You can use floating point math to compute the sample increment and lookup index, but the output must come from the lookup table. You can use either a "round to nearest" lookup strategy or linear interpolation.