

## Overview

---

This document provides a framework for lab work which can be done as a basis for the final project, described in a separate document. Some students have a hard time defining a good final project and have requested some guidance on what they should attempt. Unlike Labs 1-4, specific elements are not dictated in this lab assignment. Students are free to modify the lab assignment as they wish, so as to pursue areas of personal interest. Students are also free to follow their own final project ideas which are different from the ideas shown in this document.

This framework defines a final project which could earn an average grade. Students are encouraged to make additions to the project defined below in order to qualify for higher grades.

## Details

---

1. Refer to the final project assignment for more details about requirements and about how to select components for your project.
2. Choose a multiple-output digital-to-analog converter (DAC) and interface it to your existing hardware. Write firmware which allows the user to control each output independently. For example, the user should be able to set the output voltage of each channel. Possible options for the DAC interface include a parallel interface or a serial interface such as SPI, 3-wire, or Microwire. The user input could be from a terminal emulator window or from a keypad which is connected to your hardware. The present value of each DAC output should be displayed on the terminal emulator window or on the LCD. You should be able to verify the output voltages with a multimeter or logic analyzer. Multiple companies make DACs; some of these manufacturers are listed on the course web site. Examples of appropriate DACs are the Maxim MAX509, MAX510, MAX512 and MAX518.
3. Modify your DAC control software to provide a simple function generator. Your function generator should support square waves, triangle waves, and sine waves. The user should be able to adjust the frequency and the amplitude of the output signal. Display the waveform on an oscilloscope.
4. Choose a multiple-input analog-to-digital converter (ADC) and interface it to your existing hardware. Write firmware which allows the user to sample the signal at each input independently. The firmware should display the sampled value on the LCD or terminal emulator screen. Hook a resistor divider network (with a variable potentiometer) up to one of the ADC inputs and verify that you can sample voltages accurately by comparing the ADC value to a multimeter reading. Perhaps hook one of your DAC outputs to one of your ADC inputs. One appropriate ADC is the Maxim MAX1113.
5. Add one or more sensors to your ADC. Several companies make temperature sensors, such as National (e.g. LM34DZ) and Maxim. Check whether the part is available in a non-surface mount package, such as a TO-92 package. Verify that your firmware can calculate an accurate value based on the sensor reading. Heat and cool your temperature sensor to verify its functionality.
6. You could hook up a real-time clock (RTC) in order to allow time stamping of any measurements taken with your ADC. Manufacturers of RTCs include Dallas Semiconductor (e.g. DS1687, <http://www.maxim-ic.com>) and Epson (e.g. RTC72421B, <http://www.eea.epson.com>).