W. E. Dunn

ME 360: FUNDAMENTALS OF SIGNAL PROCESSING, INSTRUMENTATION AND CONTROL

Laboratory No. 2
Signal Conditioning and Analog-to-digital Signal Conversion Issues

1. CREDITS

Originated: N. R. Miller, July 1997
Last Updated: D. Block, September 2009

2. OBJECTIVES

(a) Become familiar with the PC-based signal conversion hardware and software available in the laboratory.
(b) Study the problems of aliasing and quantization error associated with the digital representation of analog signals.
(c) Demonstrate the noise attenuation techniques of filtering and integration.

3. KEY CONCEPTS

(a) Information is irreversibly lost when an analog signal is converted to digital form. The loss is minimized by sampling at a high rate and using a high-resolution digital representation.
(b) A digitized sine wave appears to be a sine wave at a lower frequency if the sampling rate is less than twice the frequency. This effect, known as aliasing, distorts the frequency spectrum of any signal by mapping higher frequency components into lower frequency components.
(c) Aliasing cannot be detected by examining the digitized signal alone.
(d) Thermocouples measure temperature by producing millivolt-level signals. The leads of a thermocouple act as an antenna to pick up electrical noise from the environment. The amplitude of this noise can cause problems when measuring the output of the thermocouple.

4. SYNOPSIS OF PROCEDURE

(a) Observe the effect of aliasing using the function generator, oscilloscope, audio earphones, PC-based data acquisition hardware, and MATLAB software.
(b) Observe quantization error in a ramp function using the PC-based digital-to-analog conversion hardware.
(c) Investigate the use of a simple, RC, low-pass filter to attenuate high-frequency noise.
(d) Measure the output of a Type-T thermocouple using the digital multimeter.

5. PROCEDURE

Important General Information – Please Read Carefully

(a) ***Always turn off the power supplies when changing connections. Dangling leads can easily contact the metal tabletop creating a short, blowing a fuse, creating an unsafe situation, or damaging the equipment.***

(b) Disconnect the leads from the instruments when not in use.

(c) If your station is missing something, ask your Laboratory Assistant to replace it. Do not take items from other stations.

5.1 Power On the Computer and Start the MATLAB Software

(a) Turn on the power to your station, power on the computer, and start the MATLAB software.

5.2 Sine-wave Reconstruction Using Sampled Data

Overview of Procedure

The purpose of this part of the experiment is to illustrate the effect of aliasing. Aliasing occurs when the sampling rate of a periodic signal is less than twice the frequency of the signal. For a 1000-Hz sine wave, aliasing occurs if the sampling rate is less than 2 $\times$ (1000 Hz) = 2000 samples per second. Aliasing is seen in the reconstructed waveform, which appears to be sine wave at a lower frequency. For example, consider an 1100-Hz sine wave sampled at 2000 Hz. The reconstructed signal will appear to be a sine wave at 900 Hz. Aliasing alters the observed frequency spectrum of a signal by mapping higher frequency components into lower frequency components.

The effect of aliasing on spectrum analysis can be reduced by passing the signal through an anti-aliasing filter before it is digitized. An anti-aliasing filter is a high-order, analog, low-pass filter with a cutoff frequency that is half the sampling rate. The anti-aliasing filter attenuates the troublesome high-frequency components of the signal. For example, consider again the case of an 1100-Hz signal sampled at 2000 Hz. The signal will be aliased appearing to be a sine wave at 900 Hz. The proper anti-aliasing filter will have a cutoff frequency of 1000 Hz. If possible, the better solution is to sample at a rate sufficiently high to resolve all components of the signal. In this case, we should sample at a rate greater than 2200 samples per second.

To demonstrate the effect of aliasing, we shall produce a sine wave at a specified frequency using the function generator. We shall then set up the PC-based system to sample the signal at 2000 samples per second with an analog to digital converter (ADC) and simultaneously output the sampled signal by sending the digital data back out through a digital-to-analog converter (DAC). This approach will allow us to make an easy and direct comparison between (a) the original signal from the function generator and (b) the reconstructed signal from the PC-based system. We shall visually compare the two signals by displaying the original waveform on Channel 1 of the oscilloscope and the reconstructed waveform on Channel 2 of the oscilloscope. We shall also use audio headphones to hear the difference. The original signal will produce a tone in the left earphone, and the reconstructed signal will produce a second tone in the right earphone. Aliasing will be evident when the tones are different.

We shall begin with a sine wave at 100 Hz. Because the sampling rate (2000 samples per second) is much greater than two times the frequency of the signal (2 $\times$ 100 Hz = 200 samples per second), the reconstructed signal will be a very good approximation of the original signal. We shall then increase the frequency of the sine wave using the controls of the function generator. Aliasing will begin when the frequency of the sine wave reaches half the sampling rate or 1000 Hz. At 1200 Hz, for example, the reconstructed waveform will appear to be sine wave at 800 Hz, and the reconstructed tone will be correspondingly lower in pitch than the original tone. The disparity will increase as the frequency increases between 1000 Hz and 2000 Hz. At 2000 Hz, the frequency of the reconstructed waveform is 0 Hz which is inaudible and also difficult to display on the oscilloscope. From 2000 to 3000 Hz, the frequency of the reconstructed waveform increases again from 0 to 1000 Hz. From 3000 Hz to 4000 Hz, the apparent frequency of the reconstructed waveform decreases from 1000 Hz to 0 Hz just as it did between 1000 and 2000 Hz. This pattern of increasing and decreasing frequency of the reconstructed signal repeats as the frequency of the original signal continues to increase. We shall verify this behavior visually using the oscilloscope and aurally using the earphones.

5.2.1 Power on Oscilloscope, Function Generator, and Digital Multimeter

(a) Power on the (i) oscilloscope, (ii) function generator, and (iii) digital multimeter.

5.2.2 Program Function Generator for a 10 Vp-p, 100-Hz Sinusoid with 0 VDC Offset

(a) *As when always using this instrument in the lab, first recall state "3" to initialize the function generator to its default "lab" state!*

(b) Program the function generator to produce a 100-Hz sine wave with 10 Vp-p amplitude and 0 VDC offset.

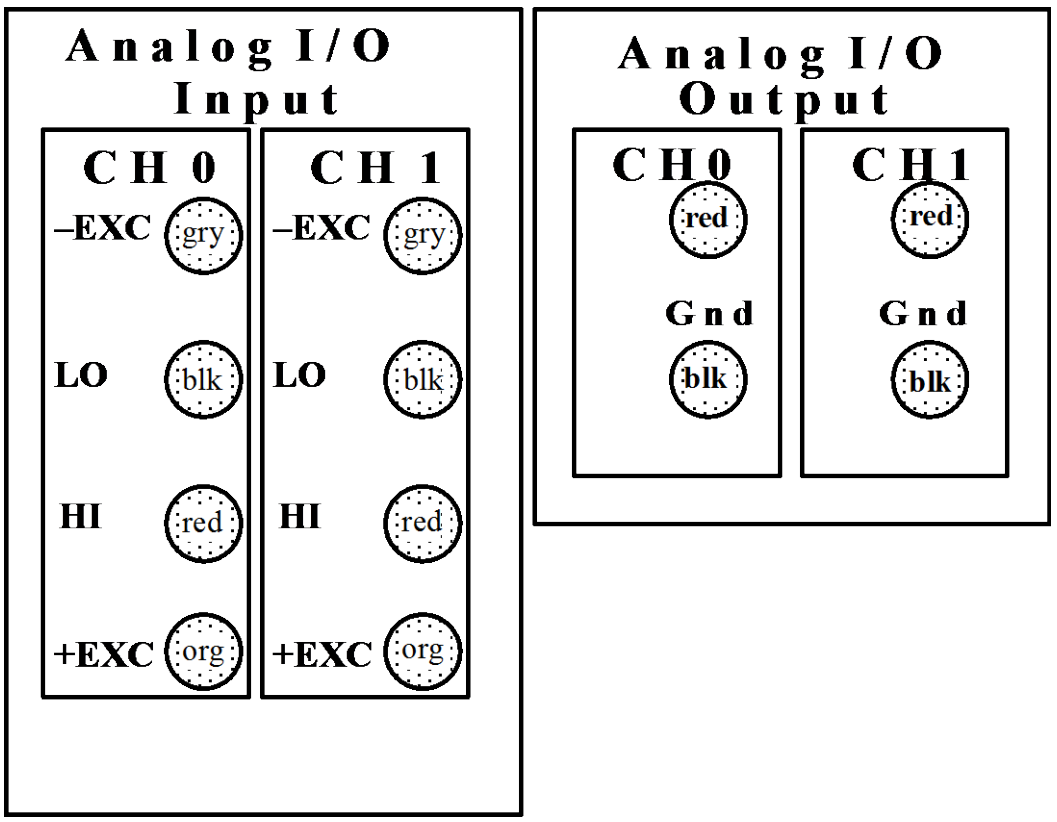
5.2.3 Wire the System to Display the Function Generator Signal on Channel 1 and the Reconstructed Waveform on Channel 2; Attach the Earphones

(c) Make sure the power to the patch panel is off and then wire the system as shown in Fig. 1.

Detailed Procedure for Wiring System for Waveform Reconstruction Experiment

(i) Use the toggle switch on the left-hand side of the patch panel to turn the power off. The red indicator light should be out.

(ii) Locate the Analog Input and Analog Output sockets on the patch panel. Note that there are two Analog Input Channels (0 and 1) and two Analog Output Channels (0 and 1) as shown below.



(iii) Find three leads each with a pair of banana plugs on one end and a BNC connector on the other end.

(iv) Wire the output of the function generator to Analog Input Channel 0 of the patch panel. To do this, connect a lead from the function generator (BNC) to Analog Input Channel 0 on the patch panel. The red banana plug should be inserted into the red socket labeled "HI". The black banana plug should be inserted into the black plug labeled "LO". This action allows the output of the function generator to be digitized by the PC-based hardware.

(v) Route the output of the function generator to Channel 1 of the oscilloscope as well. To do this, connect a second lead from Channel 1 of the oscilloscope (BNC) to Analog Input Channel 0 on the patch panel plugging into the banana plugs installed in Step (iv). This action allows the output of the function generator to be displayed on Channel 1 of the oscilloscope.

(vi) Wire the output of the digital-to-analog converter to Channel 2 of the oscilloscope. To do this, connect the third lead from Analog Output Channel 0 on the patch panel (banana plug) to Channel 2 of the oscilloscope (BNC). This action allows the reconstructed signal to be displayed on Channel 2 of the oscilloscope.

(vii) Locate the small circuit board and earphones at your station.

(viii) Find two red leads and two black leads with banana plugs on each end.

(ix) Route the *original signal* from Analog Input Channel 0 to the left speaker of the earphones. To do this, attach one red lead from the "HI" (red) socket of Analog Input Channel 0 on the patch panel to the blue connector on the circuit board. Attach one black lead from the "LO" (black) socket of Analog Input Channel 0 to the black connector on the circuit board.

(x) Route the *reconstructed signal* from Analog Output Channel 0 to the right speaker of the earphones. To do this, attach the second red lead from "HI" (red) socket of Analog Output Channel 0 on the patch panel to the white connector on the circuit board. Attach the second black lead from the "LO" (black) socket of Analog Output Channel 0 to the black connector on the circuit board through the black lead installed in Step (ix) above.

5.2.4 Program the Oscilloscope to Display Both the Original and Reconstructed Waveforms

(d) Program the oscilloscope to display the original waveform on Channel 1 and the reconstructed waveform on Channel 2 using a 5-V/div vertical scale and a 2-ms/div time base. Note that it may be necessary to adjust the oscilloscope controls for better viewing at various times during the course of the experiment.

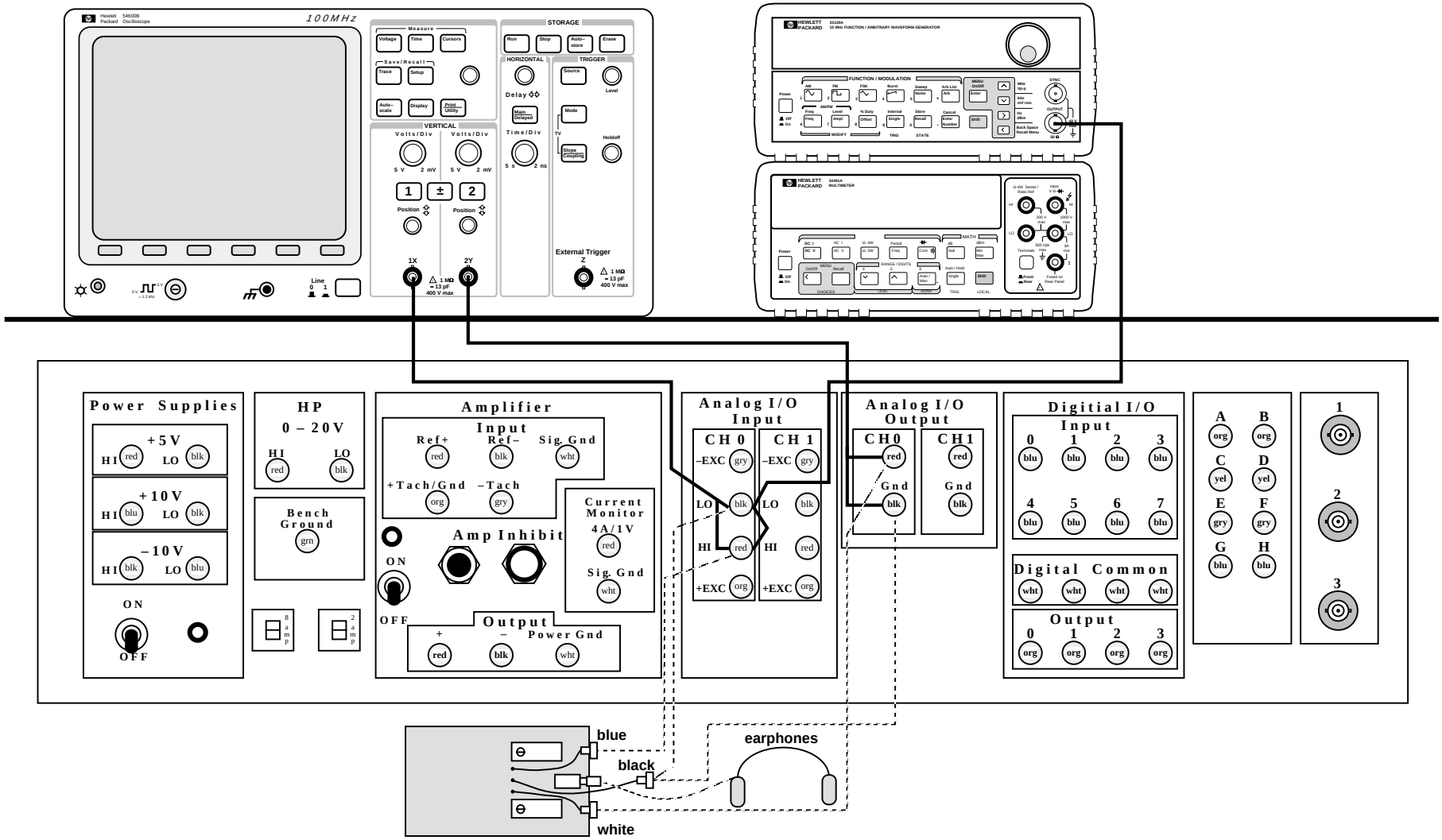


Figure 1. Equipment layout and connections for waveform reconstruction experiment.

Detailed Procedure for Programming Oscilloscope

(i) *First recall the default setup for the oscilloscope by pressing the "Save/Recall" button and the "Default Setup" soft key!*

(ii) Use buttons "1", "2", "Main/Delayed", "Mode/Coupling" and "Edge" to turn both channels on and verify they are in DC coupling mode. Set the "Normal" horizontal mode, trigger mode to "Auto" and "DC" coupling and trigger edge source to "1" with positive slope.

(iii) Set the vertical scale for both Channel 1 and Channel 2 to 5.00 V/div using the "Volts/div" knobs on the front panel of the oscilloscope. Adjust the position knobs so that the reference ground of both waveforms is the same. Recall that the reference ground is shown along the left edge of the display grid. The vertical position can be adjusted as needed later for better viewing of the waveforms. When comparing the two waveforms, it is helpful to position both channels directly on top of one another.

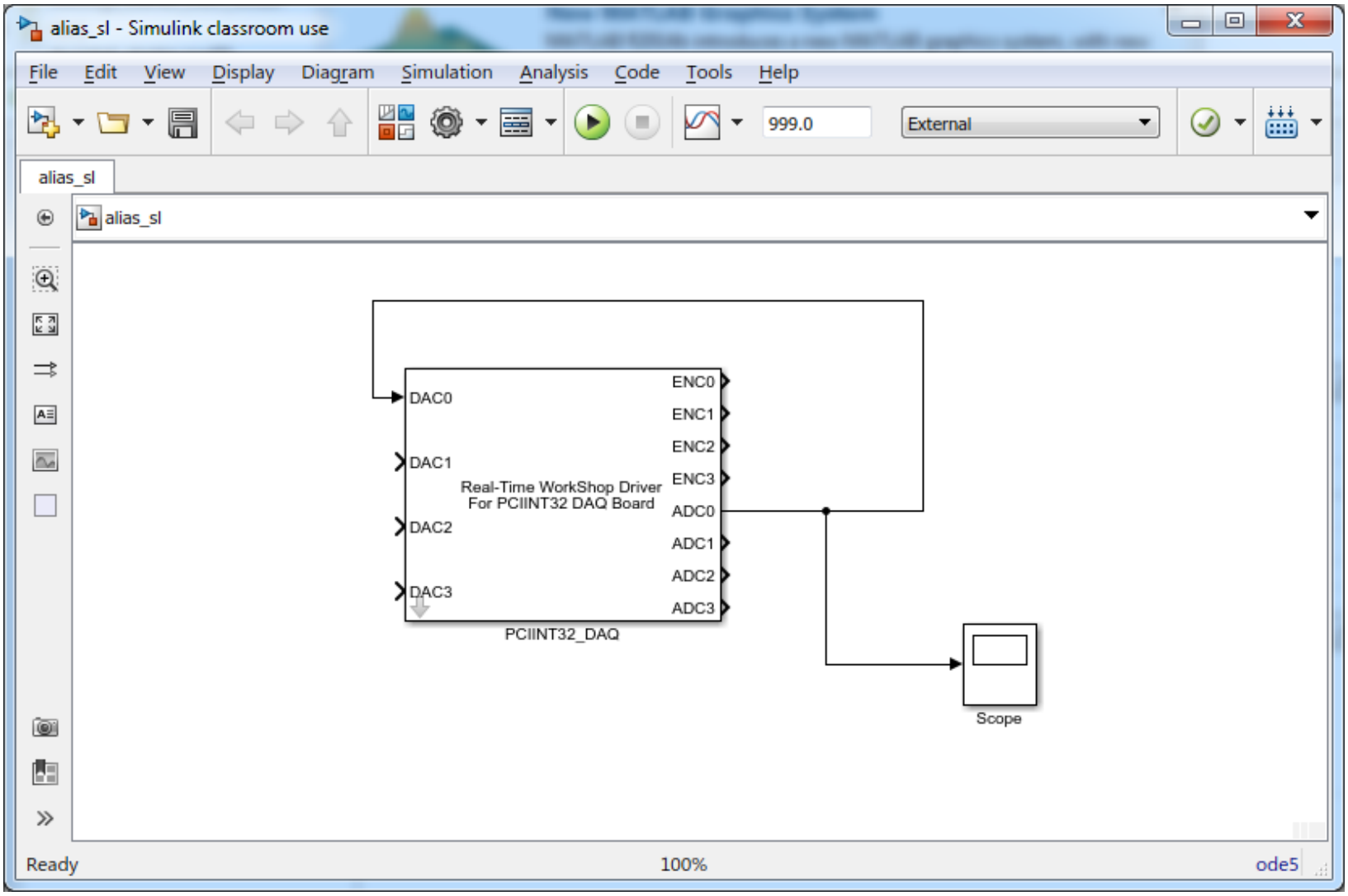
(iv) Set the horizontal time base to 2 ms/div using the "Time/div" knob. Channel 1 should show approximately two periods of a steady sinusoidal waveform. Verify that the amplitude and frequency correspond to the values programmed into the function generator. Channel 2 should show a horizontal line at 0 VDC because the PC has not yet been programmed to output the reconstructed waveform.

5.2.5 Program the Computer to Digitize and Simultaneously Reconstruct the Waveform

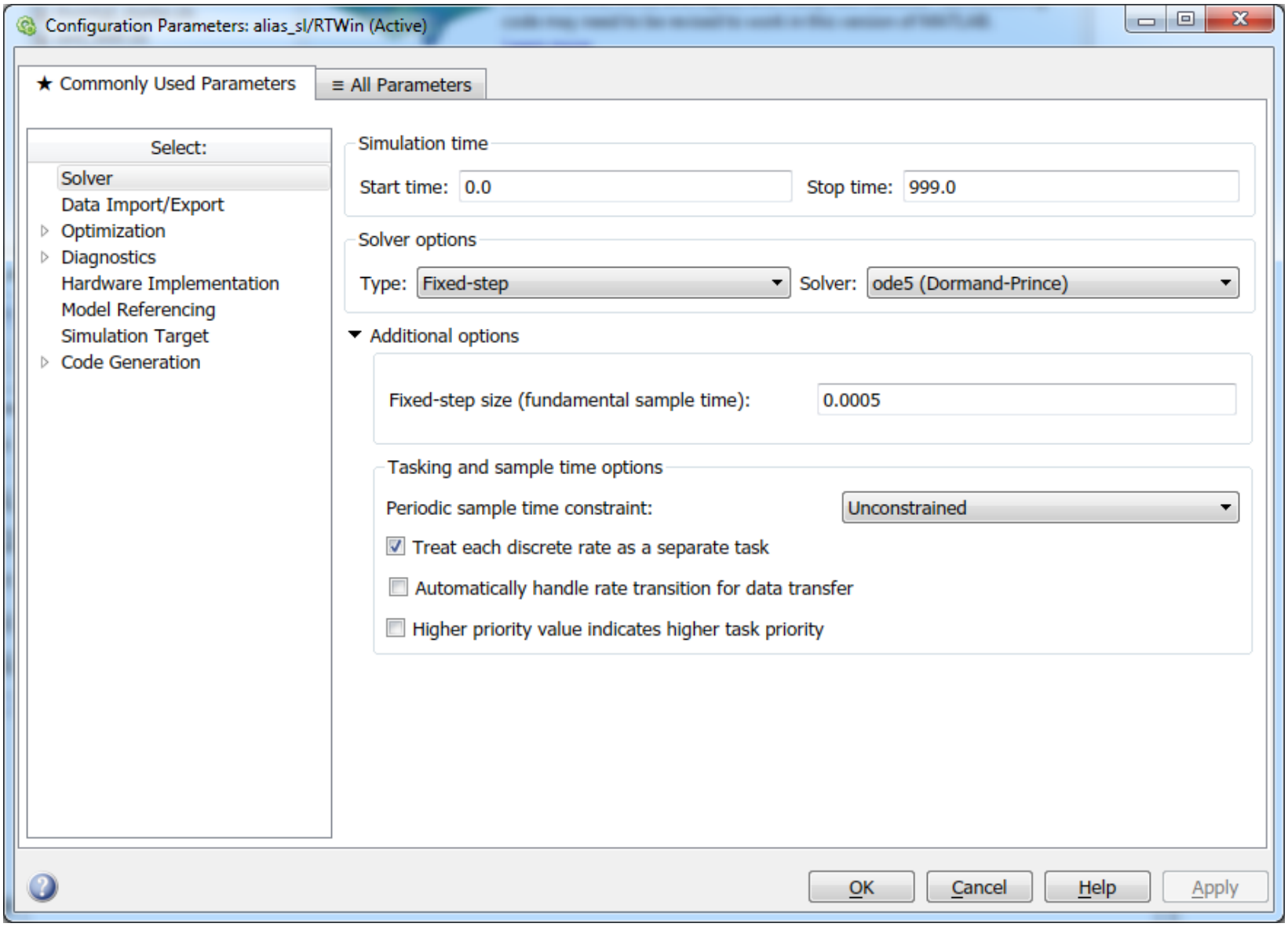
Access SIMULINK Model

(e) Enter "alias_sl" (for "alias" SIMULINK model) at the MATLAB command prompt to bring up the SIMULINK Model shown below.

The model is actually quite simple. The value sampled at analog input ADC0 is routed directly back to the analog output DAC0. These analog inputs and outputs are wired to the front panel of your lab bench. See Appendix A.



Set Analog-to-digital Sampling Rate to 2000 Hz



(f) Choose "Model Configuration Parameters" under the "Simulation" menu and select “Solver” to bring up the window above.

(g) If needed, enter "0.0005" in the "Fixed step size" text box. Then click "OK". This action sets the time step Δt between successive samples to 0.0005 s. The sampling rate $f_s = 1 / \Delta t$ is then 2000 samples per second.

Create Personal Directory on C: Drive

(h) At the MATLAB command prompt, enter “cd C:\matlab\me360" to change the current working directory.

(i) Enter "cd" to show the active directory. Make sure it is "C:\matlab\me360".

(j) Enter "mkdir *your_directory*" to make a personal subdirectory. Use the NetID of one of the group members in place of the words "*your_directory*".

(k) Enter "cd *your_directory*" to change to your personal subdirectory where again "*your_directory*" is the name used in Step (j) above.

Save Local Copy of Model on C: Drive

(l) The original SIMULINK model “alias_sl” is stored on a server shared by all the stations. The server appears as the N: drive on the computer at your station. To guard against possible network glitches, the model must be saved on the internal disk or C: drive of the computer at your station. This is done as follows.

(i) Choose "Save As" under the "File" menu to bring up the save dialog box.
(ii) Move through the directory structure to your personal subdirectory, "C:\matlab\me360*your_ directory*".
(iii) Save the file with a new name to your subdirectory. A new name could be something like “my_alias.slx”.

Build Real-Time Windows Target Executable Version of Model

(m) Select "Deploy to Hardware" under the "Code->C/C++ Code" menu to build an executable version of the model (or use the shortcut “Ctrl+B”). A C-code version of the model is first created, and then this code is compiled to produce the executable version. A toolbox for Matlab called “Real-Time Windows Target” runs this executable in a real-time extension for Windows.

(n) After the executable code is generated and compiled, you will see “Building” replaced by “Ready” in the lower left hand corner of the Simulink window.

5.2.6 Observe the Original and Reconstructed Waveforms on the Oscilloscope

(o) Turn on the power to the patch panel using the toggle switch at the far left.

(p) Start your Real-Time Window Target application by pressing the green “Run” button. Once the real-time code is started observe the oscilloscope display. The reconstructed waveform should now be displayed on Channel 2 along with the original waveform on Channel 1.

5.2.7 See and Hear Aliasing

Overview of Procedure

The procedure used here has two parts, one qualitative and the other quantitative. For the qualitative part, we shall put on the earphones and observe the oscilloscope display as we vary the waveform frequency using the large knob on the function generator. We shall compare the original and reconstructed waveforms on the oscilloscope. With the earphones the original and reconstructed tones can be heard and compared. Each lab partner shall independently dial the frequency up and down within the different frequency regimes listed on the Data Sheet paying special attention to the behavior near 1000 and 2000 Hz, and record his or her own observations.

For the quantitative analysis, we shall measure the frequency of the reconstructed waveform and the *apparent* number of samples per period of the *reconstructed signal*. We shall use the oscilloscope to do this for several different frequencies of the original waveform produced by the function generator. We shall also determine the maximum reconstruction error. We shall then determine whether (i) the original signal is aliased and (ii) the reconstructed signal offers any clues that aliasing is occurring.

Qualitative Study of Aliasing

(q) Put on the earphones. Identify the original tone in the left earphone and the reconstructed tone in the right earphone. Two sounds will be heard in the right earphone. The lower pitch tone is the reconstructed waveform of interest. The higher pitch sound comes about because the reconstructed waveform is not a continuous sine wave but rather changes abruptly (stair stepped) to a new level every time the digital-to-analog converter is updated, which is every 0.0005 s in this experiment. The reconstructed waveform is thus the superposition of the sine wave of interest and a square wave at 2000 Hz. The latter gives rise to the second sound at 2000 Hz.

Make sure you can hear the difference between the base tone and the higher pitched sound associated with the sampling frequency. Hold the left speaker away from and slightly in front of your ear as needed to improve your ability to distinguish these tones. In the following discussion, we shall ignore the higher pitch sound and present the procedure as if only one tone is present in both earphones.

(r) Increase the frequency to 200 Hz using the large knob on the frequency generator. Look at the waveforms displayed on the oscilloscope and listen for the tone in each earphone. Sketch the waveforms. Note any differences between the two waveforms and between the two tones.

(s) Dial the frequency up and down until you can fully define the behavior of the tones and waveforms between 100 and 3000 Hz. Sketch the waveforms at 200 Hz, 1000 Hz, 1950 Hz, 2000 Hz, and 2400 Hz.

(t) For each frequency, note whether aliasing is occurring. Describe the differences between the waveforms and tones that support your findings.

Quantitative Study of Aliasing and Waveform Reconstruction Error

(u) Set the frequency of the function generator to the value shown on the Data Sheet.

(v) Measure the frequency of the reconstructed waveform f_r. using the scope’s cursors. *Make sure the cursor source is Channel 2*.

(w) Press the "Run/Stop" key on the top right-side of the oscilloscope panel to freeze the waveforms on the screen. Press "Run/Stop" again to continue tracing the waveforms.

(x) Determine the actual and apparent samples/period. The actual samples/period is given by $s = f_S / f$, where f_S is fixed at 2000 samples per second and f is the frequency of the original waveform. The apparent sampling rate s_a is determined by turning off the original waveform (Channel 1) and counting the data samples taken over one period of the reconstructed waveform.

(y) Decide if the waveform is aliased.

(z) Investigate whether aliasing can be detected from observation of the reconstructed waveform alone.

(aa) Calculate the worst-case error (absolute value only) between the original and reconstructed waveforms. The maximum absolute error in the reconstructed waveform is $E_{max} = \max [V_{original}(t) - V_{reconstructed}(t)]$. To get a good measurement of the maximum difference, you may need to stop the waveform on the scope a number of times.

5.2.8 Clean Up

(bb) Stop your "alias_sl" program by pressing the “Stop” button located next to the “Run” button.

(cc) Disconnect the earphones and circuit board from the patch panel. Turn off the function generator and remove the cable between the function generator and the patch panel. Also remove the cable connected to Channel 1 of the oscilloscope. Leave the cable between Analog Output Channel 0 and Oscilloscope Channel 2 in place.

5.3 Quantization Error in a Ramp Function

Overview of Procedure

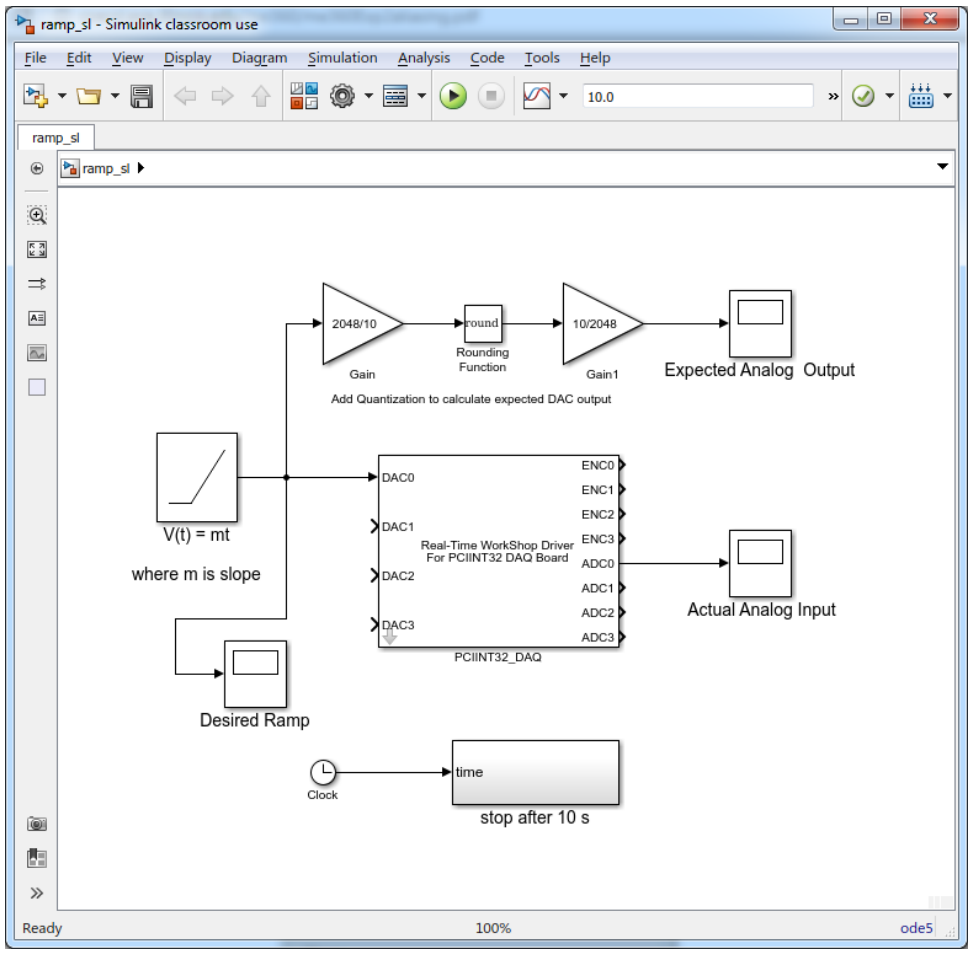
In this part of the experiment, we shall illustrate quantization error, which is the possible error between the analog voltage and the digital representation of that voltage. Quantization error occurs because only discrete or integer values exist in a digital representation. Quantization error is the result of *amplitude discretization* and may be contrasted with aliasing, which is the result of *time discretization*.

We shall use a slightly contrived situation here. We shall program the computer to produce a digital ramp from 0 to 25 mV over a 10-s period. Because the resolution of the digital-to-analog converter is 4.88 mV (See Appendix A), the analog output will actually consist of five discrete steps of 4.88 mV rather than a smooth linear ramp.

5.3.1 Program the Computer to Output a Ramp from 0 to 25 mV Over a 10-s Period

Access Ramp SIMULINK Model

(a) Enter "ramp_sl" (for "ramp" SIMULINK model) at the MATLAB command prompt to bring up the SIMULINK Model shown below.

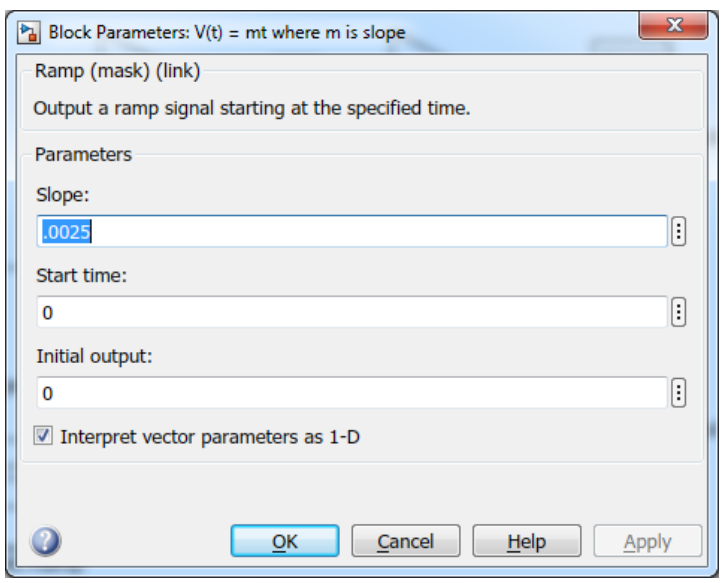


The model routes a ramp to DAC0. Simultaneously, ADC0 is sampled and saved. The clock is used to automatically stop the program after 10 s.

Set Ramp Amplitude to 25 mV

(b) Double click the ramp block labeled "V(t) = mt" to bring up the window shown to the right.

(c) Enter "0.0025" to set the slope of the ramp to 0.0025 V/s or 2.5 mV/s. Then click "OK".



Save a Local Copy of the Model on C: Drive

(d) The original SIMULINK model “ramp_sl” is stored on a server shared by all the stations. The server appears as the N: drive on the computer at your station. To guard against possible network glitches, the model must be saved on the internal disk or C: drive of the computer at your station. This is done as follows.

(i) Choose "Save As" under the "File" menu to bring up the save dialog box.
(ii) Move through the directory structure to your personal directory, "C:\matlab\me360*your directory*".
(iii) Save the file with a new name to your subdirectory. A new name could be something like “my_ramp.slx”.

Build Real-Time Windows Target Executable Version of Model

(e) Build an executable version of the model. A C-code version of the model is first created, and then this code is compiled to produce the executable version.

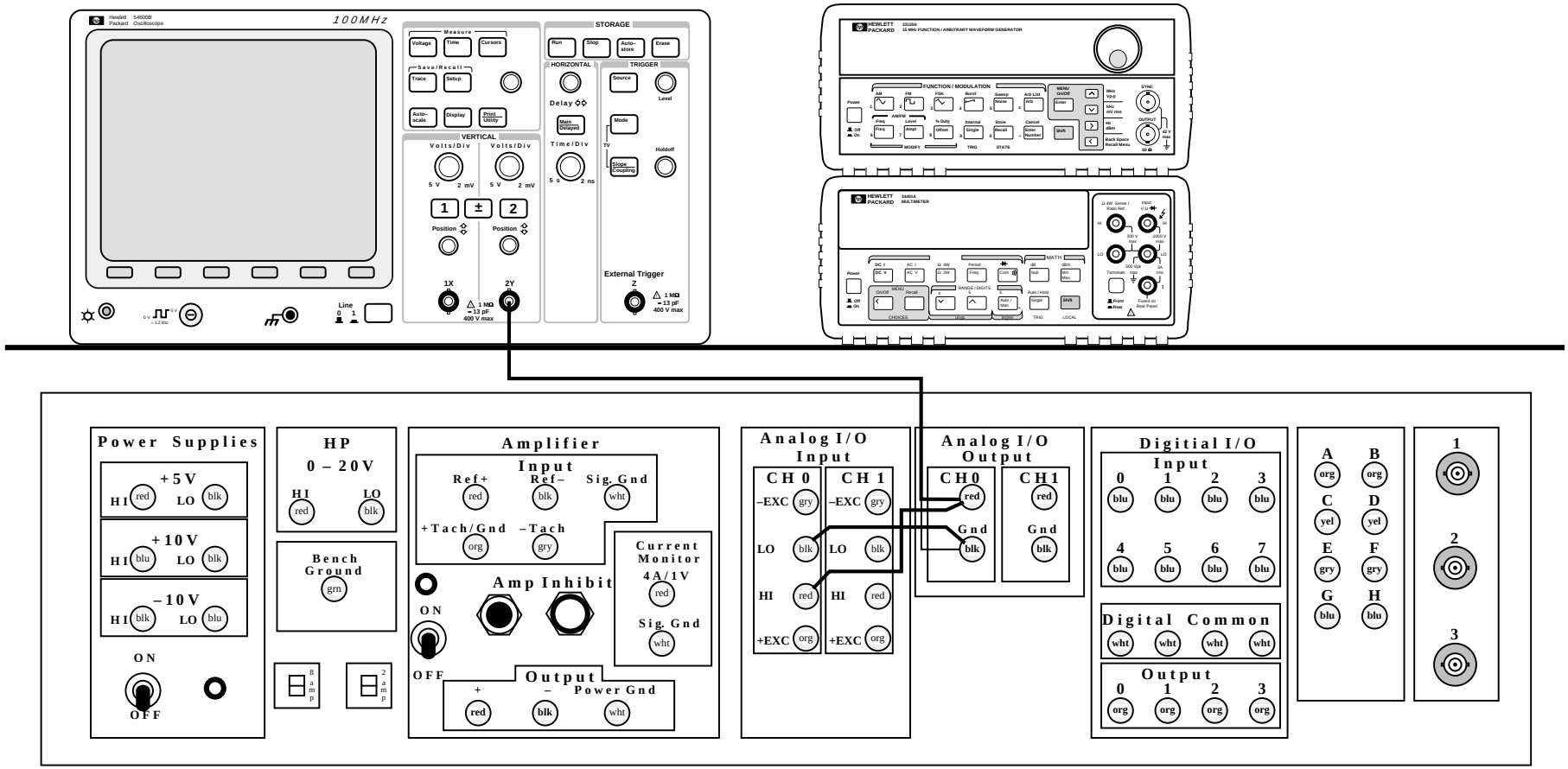


Figure 2. Equipment layout and connections for quantification error experiment.

5.3.2 Wire the Hardware to Display and Sample the Analog Ramp Output of the D/A Converter

(g) Wire the system as shown in Fig. 2. Connect Analog Output Channel 0 to both Analog *Input* Channel 0 and Oscilloscope Channel 2.

Detailed Procedure for Wiring System for Ramp Experiment

- The connection from Analog Output Channel 0 to Channel 2 of the oscilloscope should still be in place. Therefore, it is only necessary to jumper the analog output to the analog input on the patch panel.
- Take a pair of leads with banana plugs on both ends to jumper Analog Output Channel 0 to Analog Input Channel 0 on the patch panel.
- The red output socket should be connected to the red input socket, and the black output socket should be connected to the black input socket. *Note that red is on top for the output channel and on the bottom for the input channel.*

5.3.3 Program the Oscilloscope to Observe the Analog Ramp Output

(h) Using the “Main/Delayed” button, put the oscilloscope in "roll" mode. Set the time base to 1 s/div and the voltage scale to 5 mV/div.

5.3.4 Measure the Resolution and DC Offset of the D/A Converter from the Analog Ramp Output

(i) Start your Real-Time Window Target application by pressing the green “Run” button. Once the real-time code is started observe the Analog Ramp Output on the oscilloscope.

(j) The ramp output should be swamped by high frequency noise. To reduce this noise, find the low-pass, RC filter at your station. The filter is in a small blue box with a BNC connector on each end. Insert the filter between the Channel 2 input of the oscilloscope and the BNC connector on the lead from the patch panel.

(k) Run the program again and observe the Analog Ramp Output on the oscilloscope. Press the "Run/Stop" key on the front panel of the oscilloscope just as the waveform completely fills the screen.

(l) Now reset the vertical scale to 2 mV/div to take advantage of the higher resolution even though not all of the waveform will fit on the screen. You will need to run the program twice to measure all the steps.

(m) Remember to press the "Run/Stop" key on the front panel of the oscilloscope before re-running the program.

(n) Use the oscilloscope cursors to measure the resolution of the D/A converter (i. e., the step height). Repeat the measurement for all five steps and compute the average.

(o) The baseline or zero output of the ramp may not correspond to 0 V on the oscilloscope owing to a DC offset in the output voltage from the D/A converter. Measure this offset and record it on the Data Sheet.

5.3.5 Plot of the Digital Ramp Input, Analog Ramp Output, and Analog Ramp Input

(p) You should have also noticed that the Scope windows were plotting data during the run. Run the real-time application again an observe what is being plotted in the graphs. Make sure you understand each trace in the Scope windows.

(q) Answer the two questions on the data sheet for this section before continuing.

5.3.6 Clean Up

(r) Close your ramp_sl simulink file. Turn off the power to the patch panel. Remove all leads from the patch panel and the oscilloscope.

5.4 Measuring a Thermocouple Output with the DMM and Oscilloscope

A thermocouple is a simple device for measuring temperature that consists of two wires of different metals welded together to form a measurement junction. Owing to the difference in electrochemical potential between the two metals, a small temperature-dependent voltage is produced. The thermocouple used in this experiment is Type-T meaning that the two wires are copper and Constantan®, a copper-nickel alloy. When the leads of a thermocouple are attached to an instrument for measuring the small voltage, two additional thermocouples are created at the points of contact between the instrument inputs and the thermocouple leads. For our purposes, it suffices to say that the temperature at the measurement junction T_{meas} can be approximated by the following simple relationship.

$$T_{meas} = T_{ref} + V_{TC} / E$$

$$E = 41.2\ \mu V/°C$$

where T_{ref} is the temperature at the reference junctions (where the leads plug into the meter), V_{TC} is the voltage generated by the thermocouple, and E is the Seebeck coefficient which we shall take to be 41.2 µV/°C for Type-T thermocouples in the temperature range considered here. The temperature of the reference junction will be taken as the ambient air temperature as measured by the thermostat next to the hallway door.

5.4.1 Measure Thermocouple Output Using the DMM

(a) Record the ambient air temperature from the thermostat next to the hallway door.

(b) Connect a cable with a BNC connector on one end and a pair of banana plugs on the other end between oscilloscope Channel 1 and the voltage inputs of the DMM.

(c) Locate the thermocouple at your station. The thermocouple should have red and blue banana plugs. The leads of the thermocouple follow a color-coding scheme that is standard for thermocouples and not the red/black convention commonly used in electronics. **For a Type-T thermocouple, the positive wire is copper which is color-coded blue. The negative wire is Constantan® which is color-coded red.**

(d) Connect the thermocouple to the voltage inputs of the DMM *through* the banana plugs just installed. **The negative-polarity (red) Constantan® wire should be connected to the "LO" or black input of the DMM. The positive-polarity (blue) copper wire should be connected to the "HI" or red input of the DMM.**

(e) Hold the thermocouple in your hand such that good thermal contact is made with your skin and measure your skin surface temperature using the DMM and the expression given above. Record the results on the Data Sheet.

(f) Note that internal body temperature is roughly 98.6 °F (37.0 °C) on average for the entire population but does vary slightly from person to person. The temperature of one's skin is far more variable from one person to the next and from time-to-time, depending on activity level, etc. Values in the range of 85 to 95°F (29 to 35 °C) are typical.

5.4.2 TA Demonstration of possible aliasing problems in Thermocouple measurements.

(f) Make sure to answer the question in the data sheet for this section before you leave the lab.

Appendix A. PC-Based Hardware and Software

Each laboratory station has a Windows®-based personal computer with internal hardware for performing analog-to-digital (A/D) and digital-to-analog (D/A) conversion. The basic configuration is shown in Fig. A.1.

The analog inputs from the patch panel are passed to two Analog Devices 5B41 isolation amplifiers each with a gain of 0.5. These modules map a voltage range of -10 to +10 VDC at the patch panel inputs onto the range of -5 to +5 VDC accepted by the PC-based analog-to-digital converter. The 0.5 gain of the isolation amplifier is accounted for in the data acquisition block of our SIMULINK model.

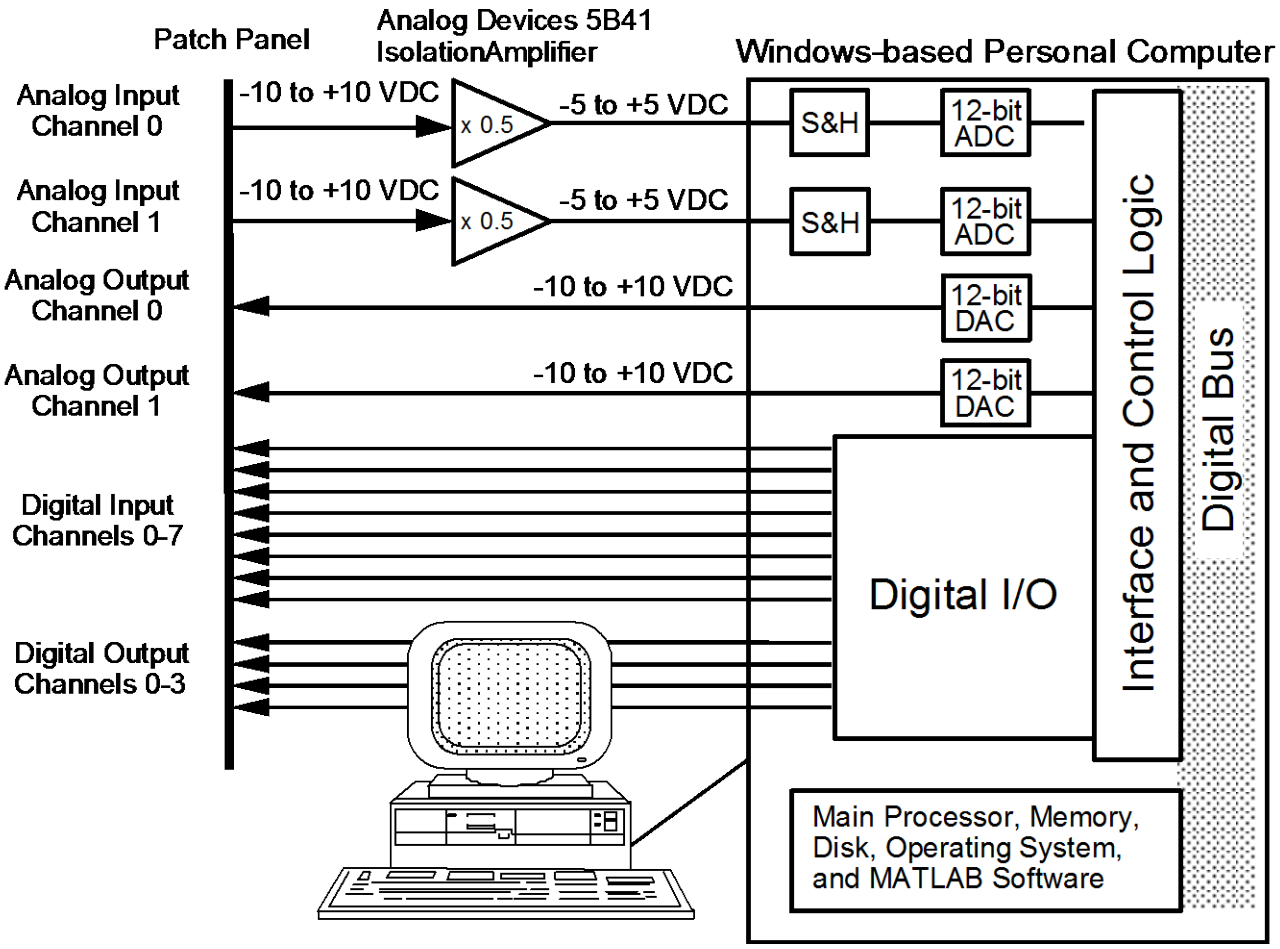


Figure A.1 PC-based Hardware for A/D and D/A Conversion.

The PC has a 12-bit analog-to-digital converter. It uses successive approximation to convert the analog input voltage to a 12-bit binary integer (i. e., an integer between 0 and 4095 or between 000000000000_2 and 111111111111_2 where the subscript "$_2$" denotes a base-2 or binary number). The sample-and-hold circuit of the analog-to-digital converter samples the input voltage at a particular instant of time and holds this value steady while the conversion is made. The digital or integer representation of the input voltage n_{in} is given by

$$n_{in} = \text{truncate}\left(2^{n_b} \frac{V_{in} - V_{min}}{V_{max} - V_{min}} \right) = \text{truncate}\left(\frac{V_{in} - V_{min}}{\Delta V_{bit}} \right) ,$$

where n_b = 12 = number of bits in the integer representation, V_{min} = -10 VDC = the minimum patch-panel input voltage, V_{max} = +10 VDC = maximum patch-panel input voltage, $\Delta V_{bit} = (V_{max} - V_{min}) / 2^{n_b}$ = 4.88 mV = the bit resolution of the conversion, and "truncate" is a function that converts a real number to an integer by dropping the decimal fraction.

Similarly, the digital-to-analog converter converts a 12-bit binary integer n_{out} to a voltage between -10 V and +10 V according to

$$V_{out} = n_{out} \frac{V_{max} - V_{min}}{2^{n_b}} + V_{min} = n_{out} \Delta V_{bit} + V_{min}.$$

The conversions are made at a software-selectable discrete time step Δt. The sampling rate f_s given by $1/\Delta t$ is expressed in units of samples per second or s^{-1}. For the analog-to-digital conversions, we assume that the sample-and-hold acquisition time is much smaller than the time step so that the sampling may be considered instantaneous.

Appendix B. Analog-to-digital Conversion Issues

When an analog signal is digitized, both time and amplitude are mapped onto a set of integers. This process is often referred to as "discretization" because the digital representation consists of discrete (integer) values. Time discretization occurs because the analog signal is sampled at particular instants or over particular intervals of time. Amplitude discretization occurs because the amplitude of the analog signal is converted to a binary integer between 0 and $2^{n_b} - 1$ where n_b is the number of binary digits or bits in the digital representation. Some information is inevitably lost when the real variables time and amplitude are mapped to integers. *Waveform reconstruction error* and *aliasing* are associated with the *discrete time* nature of the digital representation whereas *quantization error* is associated with the *discrete amplitude* nature of the digital representation. We consider reconstruction error and aliasing in Section B.1 and treat *quantization* error in Section B.2.

B.1 Waveform Reconstruction Error and Aliasing

Waveform reconstruction is the process by which the original signal is reconstructed from the digital representation. Reconstruction error is the difference between the reconstructed signal and the original signal. The magnitude of this error depends on the sampling rate and the method of reconstruction. Aliasing is a special type of reconstruction error that occurs when the sampling rate is less than twice the frequency of the original signal. Under these conditions, a reconstructed sine wave (or frequency component of a more complex waveform) appears to be at a lower frequency than it actually is.

Here, we examine three methods of waveform reconstruction: (i) histogram reconstruction, (ii) vector reconstruction, and (iii) Fourier reconstruction (See Figure B.1). These three methods are illustrated in Fig. B.1. In histogram reconstruction, the voltage level is changed at the start of each time step and then held constant until the beginning of the next time step. Most digital-to-analog converters use this method of approximation because of its simplicity.

One level up in terms of both complexity and accuracy is the vector or trapezoidal method. With the vector method, the voltage is linearly interpolated between two successive values. This method of reconstruction is used by the digital oscilloscope at your station.

The Fourier reconstructed waveform $V_f(t)$ is given by

$$V_f(t) \quad = \quad \sum_{k=-\infty}^{k=+\infty} \left\{ V_k \, \frac{\sin\left[\pi\left(\frac{t}{\Delta t} - k\right)\right]}{\pi\left(\frac{t}{\Delta t} - k\right)} \right\}$$

where V_k is the sampled voltage given by

$$V_k \quad = \quad V(k\,\Delta t) \quad .$$

Note that Fourier reconstruction requires an infinite number of samples ranging from $k = -\infty$ to $k = +\infty$. If the signal is periodic and sampled at an integer multiple of the waveform frequency, then the required values can be obtained by periodic extension. Otherwise, the data to fully implement Fourier reconstruction do not exist. Fourier reconstruction is immensely important because this method can be used to completely recover the original waveform provided that certain conditions are met. One of the conditions that has to be met is known as the Nyquist sampling theorem. *Fourier reconstruction returns the original waveform without error if the sampling rate is at least twice the highest frequency component of the waveform.* If the sampling rate is not at least twice the highest frequency than aliasing occurs. So aliasing occurs when a signal is sampled too slowly (See Figure B.2).

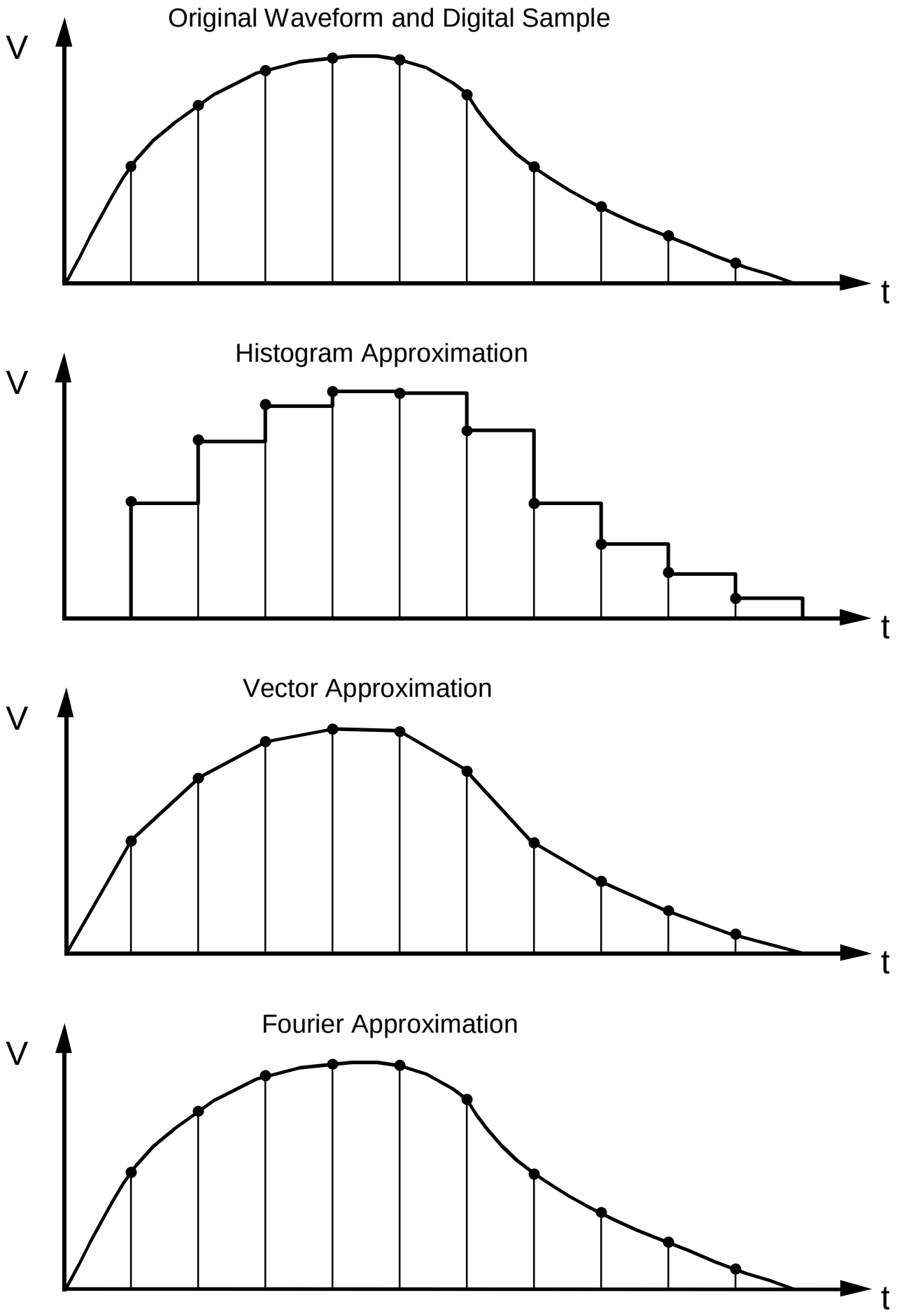


Figure B.1 Three Methods of Reconstructing a Waveform from its Digital Representation.

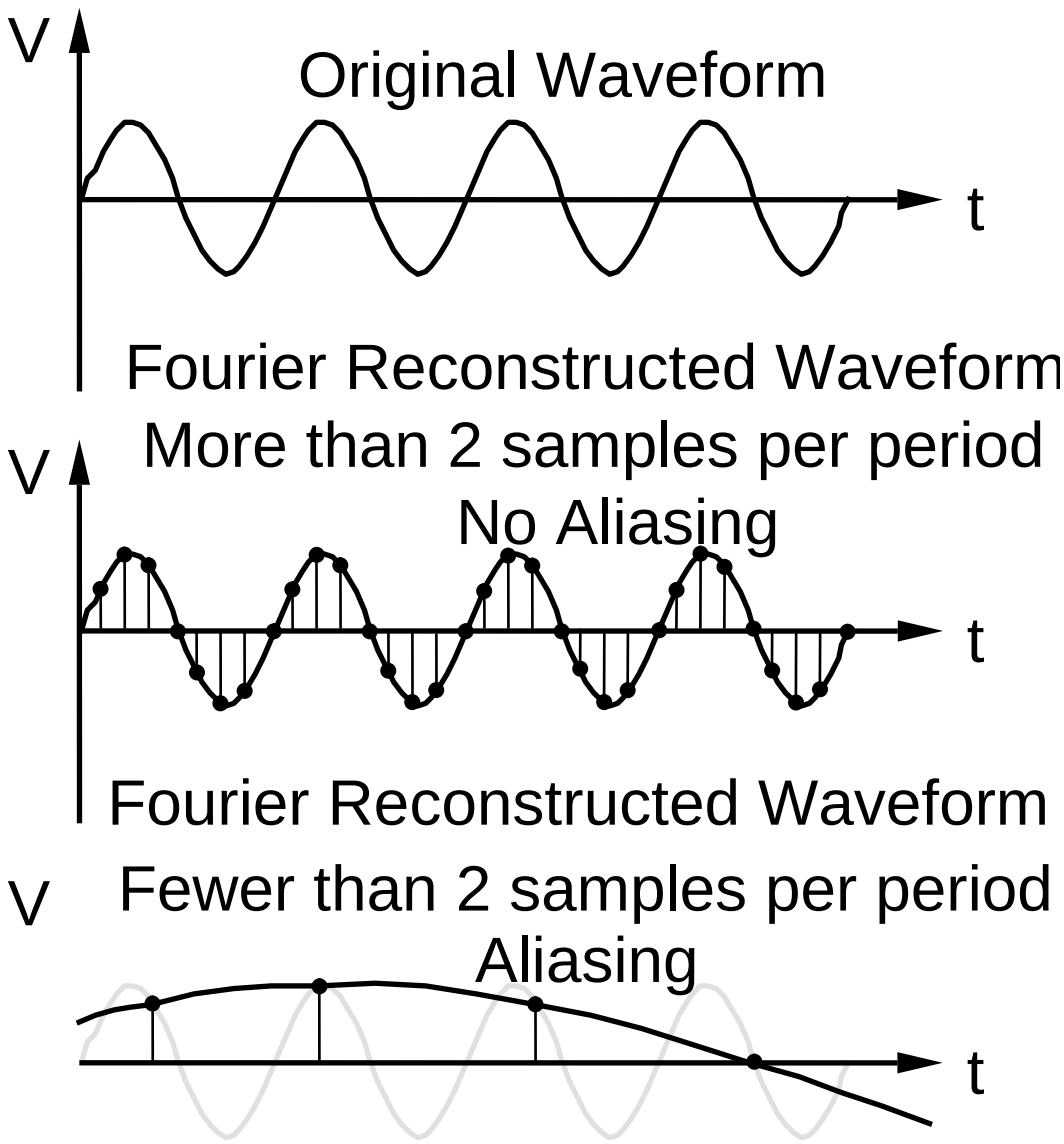


Figure B.2 Effect of Sampling Rate on Waveform Reconstruction.

Aliasing can only be reduced by sampling at a higher rate. When this is not possible, then low-pass filtering can be used to reduce the amplitude of the high frequency components and thus lessen the negative effect of aliasing on spectral analysis. A low-pass filter serving this latter purpose is called an "anti-aliasing filter".

Although a sampling rate of 2 samples per period is sufficient to avoid aliasing, such a low sampling rate is hardly sufficient to properly define a digitized waveform in most practical applications. Figure B.3 shows how reconstruction error varies with the number of samples per period for the three methods considered here. The test case is the approximation of a sinusoidal waveform, and the error is defined as the ratio of the maximum possible absolute error in the reconstructed waveform to the amplitude of the original signal.

The histogram method requires a large number of samples per cycle to produce a reasonable approximation. From Fig. B.3, we see that, even for 30 samples per cycle, the maximum error is still about 10 %. With the vector method, the maximum error drops below 1 % for 12 or more samples per cycle. The oscilloscope at your station normally uses the vector method. The vector method can be turned off by first pressing the "display" key on the front panel of the instrument and then turning "vectors off" with the context-sensitive keys below the display. The reconstructed waveform will then be shown as a series of dots.

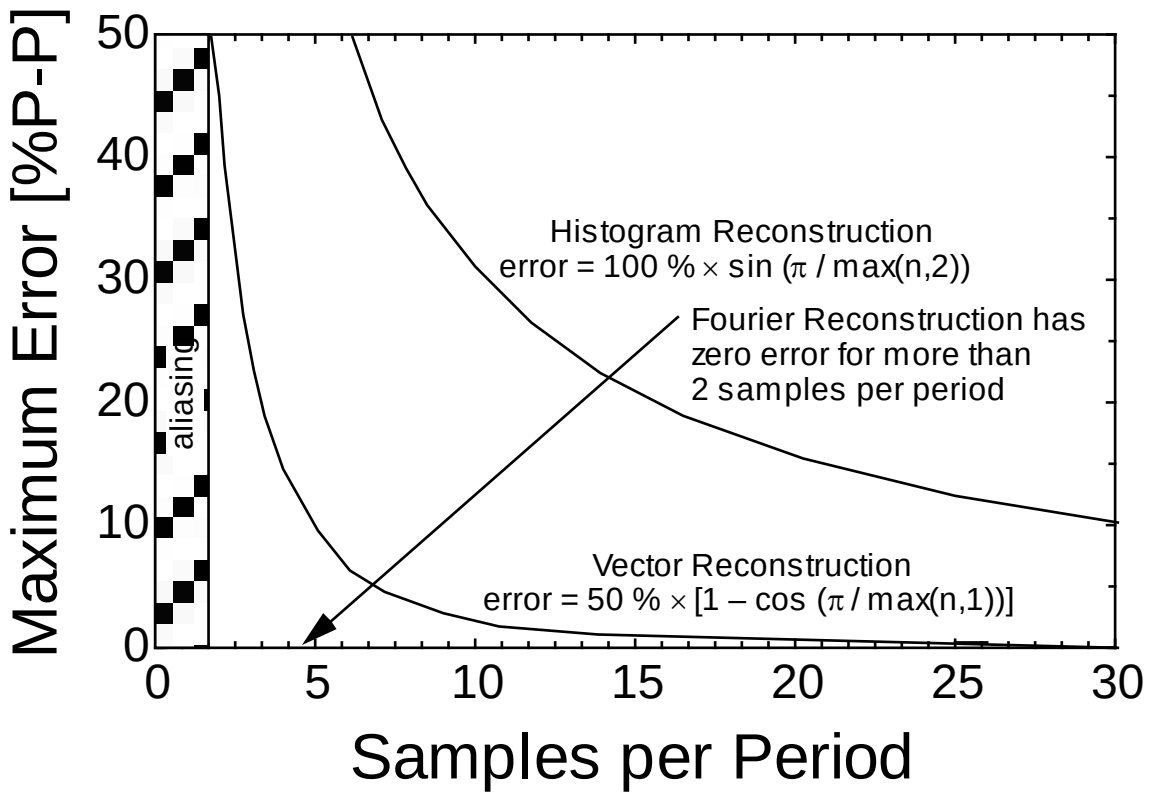


Figure B.3 Effect of Sampling Rate on Maximum Error in Reconstructed Waveform.

B.2 Quantization Error

Quantization error arises because the signal is digitized in discrete amplitude steps. Quantization error is reduced by decreasing the range of the measurement or by increasing the number of bits in the binary integer representation (e. g., converting an 8-bit A/D to a 12-bit, 16-bit or even 18-bit A/D). In general, quantization error is a much less severe problem than sampling speed and aliasing. Even an 8-bit A/D converter provides a resolution of 1 part in 256 or about 0.4 %. This corresponds to roughly 256 samples per period in terms of sampling speed. A 10-bit converter provides a resolution of better than 0.1% which is adequate for many instrumentation applications. As far as experimental error goes, this is fairly small. There are, however, some digital audio and video applications where a dynamic range of three orders of magnitude or more is needed in which case quantization error is a concern. To achieve adequate resolutuion at low levels while maintaining sufficient dynamic range using an integer representation requires a large number of bits.

B.3 Noise

Noise is often a far more serious problem than either aliasing or quantization error. The waveform seen by an instrument can be separated into three components as follows.

$$\text{waveform} = \text{signal} + \text{line noise} + \text{high frequency noise}$$

Here, "signal" represents the valuable part of the waveform. "Line noise" which emanates from AC power lines and AC powered devices is at the power-line frequency (60 Hz in the U. S.) and its harmonics (integer multiples of the power-line frequency). The "high-frequency noise" component comes from a wide variety of sources and may range in frequency from a few kHz to several MHz or even GHz. Three basic methods of dealing with noise are (a) shielding, (b) filtering, and (c) integration. We shall examine the latter two here; the issue of shielding will be treated in the next experiment.

B.3.1 Filtering

Filtering can be effective against both power-line noise and high-frequency noise although it is probably best at reducing the latter. Low-pass filters can be very effective against high-frequency noise provided that the frequency of the noise is much higher than the frequency of the signal. For example, an eighth-order, low-pass filter can be obtained at reasonable cost. If the high-frequency noise is only one decade higher in frequency than the signal, then eight orders of magnitude in reduction of the high-frequency noise is achieved. Even a first-order, passive, RC filter can be effective against high-frequency noise. Notch or band-reject filters can be used to reject power-line frequencies, but these filters typically do not suppress harmonics and they may distort the desired signal.

APPENDIX C.MATLAB Tutorial

The following MATLAB tutorial can be used to familiarize yourself with basic MATLAB syntax in preparation for the use of this software during the rest of the semester.

C.1 Basic commands and syntax

(a) The MATLAB command line can be used just like a calculator to perform calculations. The result is stored in the default variable "ans". Try the following simple commands. Enter the command shown followed by a return.

6*4+3*3-2^3/2	note that standard algebraic operator preference is used
5 * cos(2*pi)	note the use of the built-in variable "pi"
5 > 3 5 < 3	note the use of the relational operator ">"; the relational operators are: "=" equal "~=" not equal "<" less than ">" greater than "<=" less than or equal ">=" greater than or equal
3 <= 5 & ... 4 >= 1	note the use of the logical operator "and" and the continuation syntax "..." (three consecutive periods); the logical operators are "&" and "|" or "~" not xor (a,b) exclusive or
% comment line	the percent sign allows comments to be mixed in with MATLAB commands

C.2 MATLAB Variables

(b) MATLAB allows variables to be defined and used in algebraic expressions. Try the following simple examples.

a = 5; b = 3; c = a * b^2	note that each input line can contain multiple expressions separated by a semicolon
d = a/b; d	note that the semicolon also suppresses the result display after the first line; whereas a variable name on a line by itself displays its current value
whos	displays detailed information about the variables defined
clear d; whos clear; whos	"clear" deletes a specific variable or all variables if no name is specified

C.3 MATLAB Arrays

(c) The fact that MATLAB provides a natural and simple syntax for handling arrays is one of its most powerful features. Try the following simple commands. Predict the result before actually entering the command.

a = [1 3*pi sqrt(2)]	note that array elements which are enclosed in brackets "[ ]" and separated by spaces can be any algebraic expression
b = [1 2 3; 4 5 6]	creates the 2-by-3 array $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$; rows can also be separated by a return
b(2,3) = 9	sets the element in row 2 and column 3 of array b to 9
b(1,2)	displays element in row 1 and column 2 of b
b(1,1:3)	displays elements in row 1 and columns 1 through 3 of b
b(2,1:2) = [7 8]	sets elements in row 2 and columns 1 through 2 of b to 7 and 8, respectively
c = 1:0.2:2	creates a row vector starting at 1 and ending at 2 with an increment of 0.2
d = linspace(0,1,6)	creates a row vector starting at 0 and ending at 1 with 6 evenly spaced elements
e = logspace(1,3,5)	creates a row vector starting at 10^1 and ending with 10^3 with 5 elements each succeeding pair of which differs by the same factor
f = zeros(2,3)	creates an array of zeros with 2 rows and 3 columns

g = 5*ones (3,1)	creates an array with 3 rows, 1 column and all elements equal to 5
h = b'	sets h equal to the transpose of b
p = sqrt(b) / 2	computes the square root of each element in b, then divides each element by 2
q = 4 * p.^2	squares each element in p, then multiplies by 2; note the special operator element-by-element operator ".^"
r = b * h	matrix multiplication in the conventional manner
s = 2 * r^2	same as 2 * r * r; matrix multiplication of r by itself is done first, then each element of the result is doubled; r must be a square matrix
t = c + d	adds two arrays in the conventional (element-by-element) manner
u = c .* d	multiplies two arrays element-by-element yielding [$c_1 * d_1$ $c_2 * d_2$ $c_3 * d_3$ $c_4 * d_4$ $c_5 * d_5$ $c_6 * d_6$]
whos	displays information on arrays defined
clear	clears all scalar and array variables

C.4 Complex Arithmetic in MATLAB

(d) MATLAB also handles complex numbers in a natural manner. Try the following examples.

a = 3 + 4 j	"j" is used to signify $\sqrt{-1}$
b = 2 + sqrt(-9) c = exp(j * pi / 6) d = abs(a) e = angle(a) f = real(a) g = imag(a)	conventional complex functions
h = a * b p = a / b	conventional complex arithmetic
q = [1+2j 3+4j]	complex arrays are constructed from complex numbers in the natural way
whos	display information about variables
clear	clear all variables

C.5 MATLAB Plotting Capability

(e) MATLAB provides powerful plotting capability. Try the examples below.

x = 0:0.01:10;	creates a row vector of 1001 elements from 0 to 10 by increments of 0.01 *The semicolon at the end of the command line suppresses the long listing of the array elements.*
y = sin(x); z = cos(x); u = x .^ (-2.5);	creates two row vectors of 1001 elements each such that $y_i = \sin(x_i)$, $z_i = \cos(x_i)$, and $u_i = x_i^{-2.5}$ *Again, the semicolons are a big time saver.*
plot (x, y, x, z)	plots y and z versus x. Note the plot may be automatically minimized. To display a minimized plot, double click on the appropriate button in the window tray at the bottom of the screen
axis ([2 5 0 1])	modifies the axes so that xmin = 2, xmax = 5, ymin = 0 and ymax = 1. Note that the axis function takes an array of four elements as a single argument and not four scalars as individual arguments. Restore the plot to the screen and note the effect.
axis auto;	restores default axes limits
loglog (x, u)	makes a log-log plot of u vs. x
semilogx (x, u)	makes a semi-log plot with the log scale of the abscissa
semilogy (x, u)	makes a semi-log plot with the log scale of the ordinate
clear	clears all variables