ECEN-361 Lab-08-Part2: ADC / Sampling

NAME: _____

Introduction and Objectives of the Lab

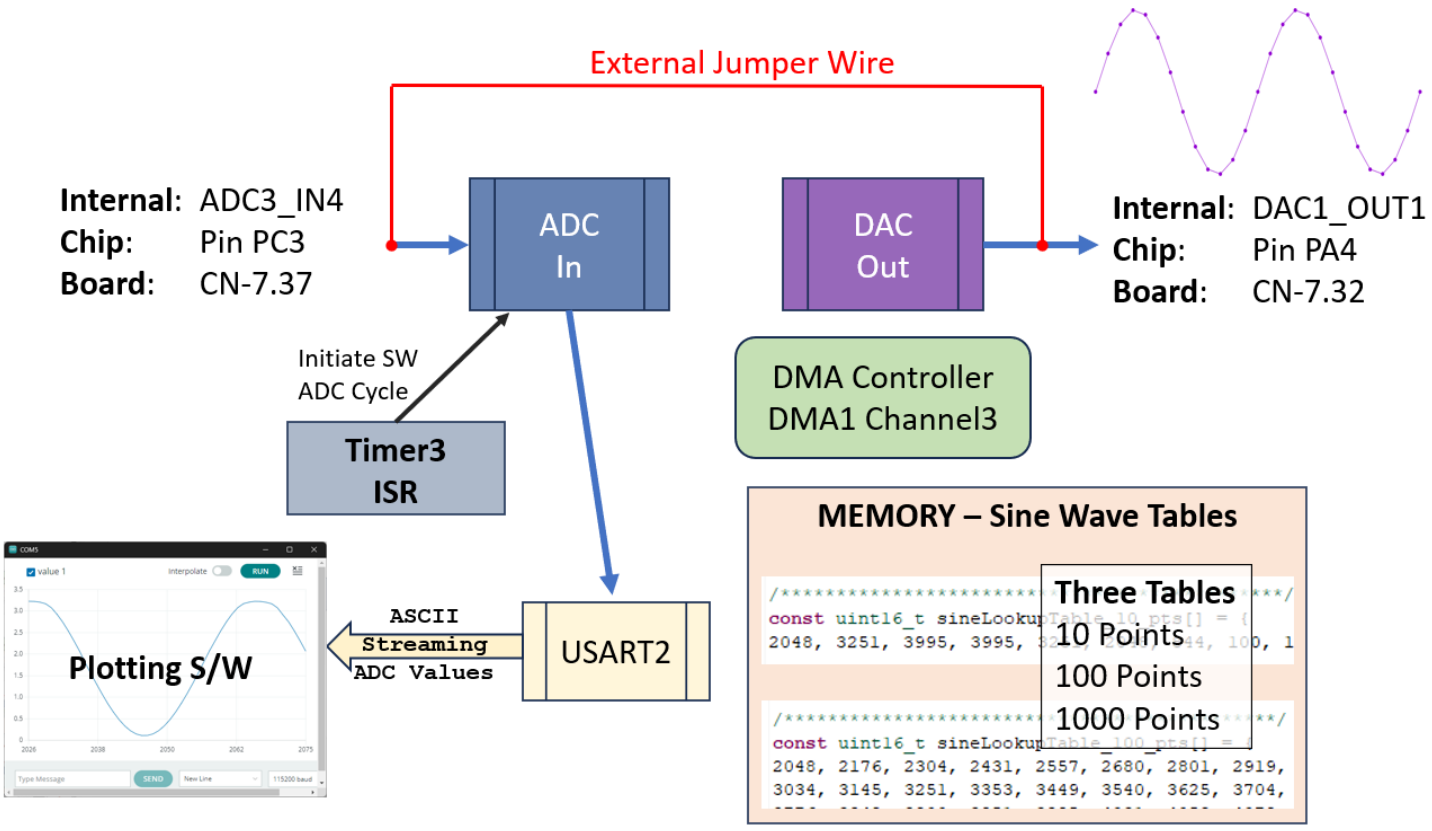
Part 1: Understand ADC operations & triggering control via timer
Part 2: Observe the effects of sampling rate on an ADC, via a Serial-port-based graphical tool

For each of the parts, follow the instructions, then fill in answers to the questions. Expected answers are indicated in the boxes with red text/spaces to fill in answers. Repositories must be pushed, code will be run.

Lab Instructions

Overview of System

This project continues from Lab-08, where the Nucleo-64 was used to create various quality sine-wave outputs via a DMA-controlled DAC, reading from tables. This project will use that same portion of the lab to provide an on-board sine wave that will then be fed back into the ADC for this portion of the lab. For this lab, the wire below will be required, as the electric signal is driven out from PA4 and sampled in with PC3:



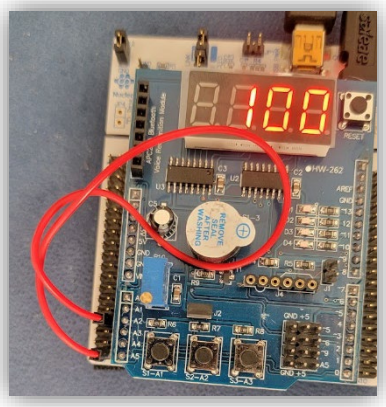
Part 1: Modifying the ADC Sample Rate and seeing

1. Accept the Assignment, Download the repo, Run the code.

Your board should show "1000" on the 7-seg display

The 7-Seg LEDs indicate which of the points-per-cycle table is being used to output the DAC waveform.:10, 100, 1000. These values can still be cycled by pressing S1. For this lab, there will be no need (maybe *Extra Credit*) to use any of the other values generated by our DAC**!**

2. Connect the DAC out to the ADC in



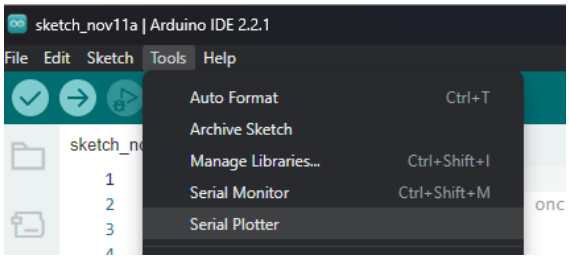
You'll need a single jumper (from your ECEN-106 kits or available during lab time).

Use a female-to-female jumper to connect the DAC-out to the ADC-in (CN-7.32 → CN-7.37).

3. Run / Use a Serial Plotter.

For this lab, you'll be looking at the sampled output of the ADC, by using software that reads ASCII values of data points and plots them. There are several free solutions. The following two have been tried and work equally well:

a. Option 1: Arduino Serial Plotter



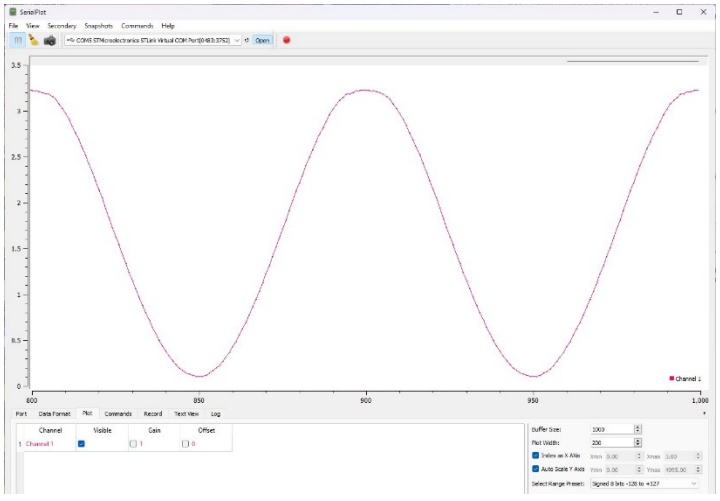
If you have already used and still have Arduino installed, this is an easy solution. Open Arduino, set the correct COM Port, and open the Tools/Serial Plotter:

b. Option 2: SerialPlot (LINK HERE)

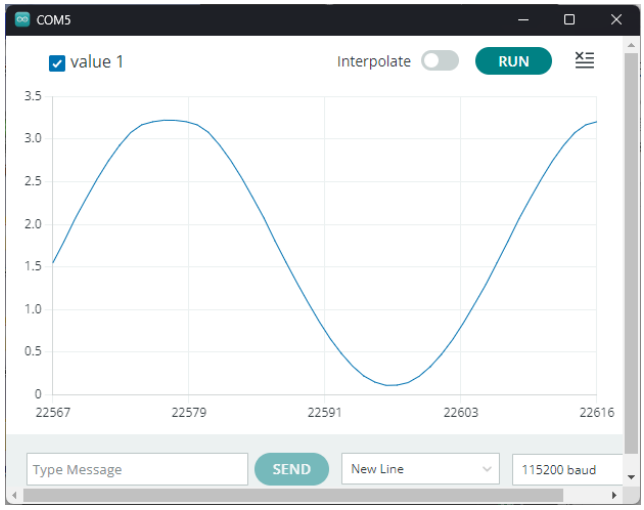
If you don't have Arduino or want to use something different (like for Mac or Linux), SerialPlot works very well. Install from the link above, connect the correct COM port, and set the "Plot Width" to about 100.

Either of these graphing tools should instantly be showing the output of the ADC:

SerialPlot



Arduino Serial Plotter



For the lab, "Serial Plot" will be shown, but results are very similar.

Note that you can see the data on the COM port as well by opening a serial terminal emulator, like PuTTY. You can't connect both PuTTY AND the plotting program at the same time.

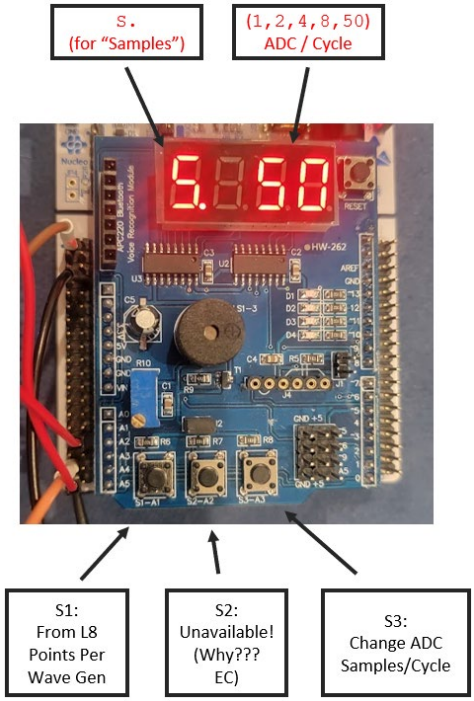
4. Sample Rate with button: S3

The on-chip SineWave Generator (at 1000pts/period) is set to output exactly 1Hz. This was shown in the last lab, and can be verified with the Saleae (or other tool), by looking at the connecting wire.

Should be clear at this point:

- The Saleae measurement is on the DAC output. It's being driven by the Nucleo-64
- We are measuring the ADC results of that signal with the graphic plotting tool

In this lab, we will change the samples per period. The SineWave source (1Hz) will stay constant, but we will sample at [1, 2, 4, 8, and 50 samples/period]. The sample rate will change via the S3 button. Pressing S3 will then change the display to show the Samples/Cycle, as "S. x" where x is the samples/period. Press S3 and see that it cycles:



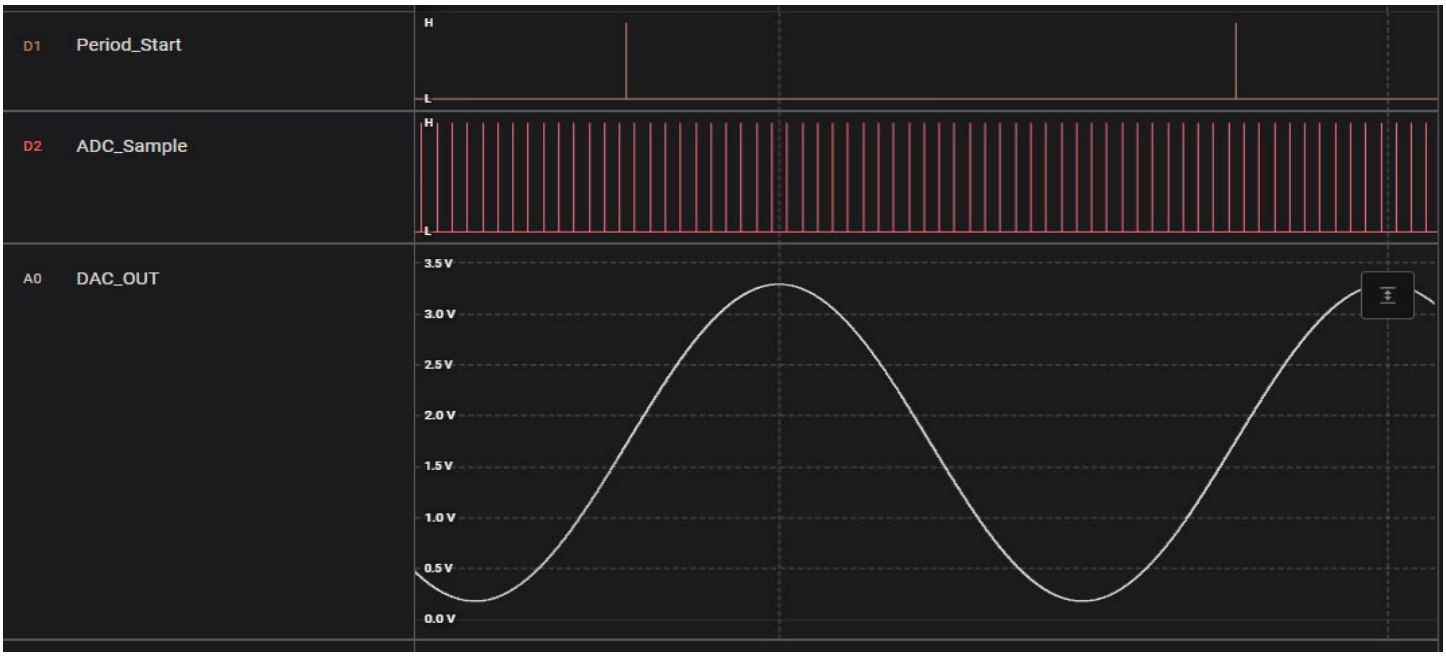
With the SerialPlotter screen connected and working, press S3 and cycle thru the samples. NOTE – Only the "8" changes things!

Computing and modifying the values necessary to create the sample rate is the exercise today.

5. Set up the Saleae Logic Analyzer

It will help to see things. Connect the following probes on the Saleae:

Channel-A0 Analog (rename to "DAC-OUT")	to the jumpered wire
Channel D1 Digital (rename to "Period_Start")	to PC10 (top left CN7.1)
Channel D2 Digital (rename to "ADC_Sample")	to PC2 (CN7.35)

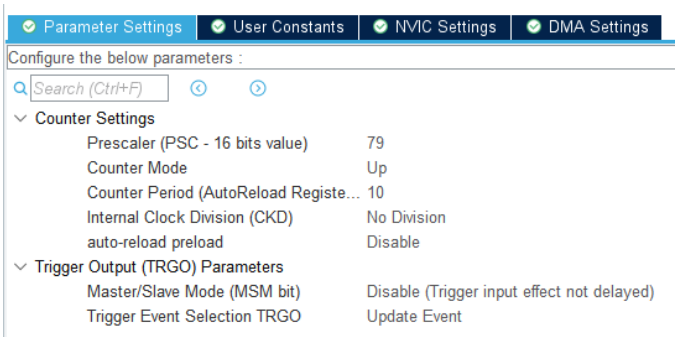


- **Period Start**: A GPIO pulse from the DAC at the beginning of the SineWave out (1000 pts)
- **ADC_Sample**: A GPIO pulse out each time an ADC sample is taken
- **DAC_OUT**: The analog wave out and going back in to be sampled

As can be seen – with 50-samples per cycle, there are 50 pulses per period.

6. Modify the code to change ADC Samples

ADC samples can be 'triggered' by various means: An external clock input, one of the internal Timer, or manually by a S/W call. Because we want to do some processing directly, we are using a Timer with the Timers ISR to manually initiate a S/W ADC. We will adjust how often that Timer ISR is called, by adjusting the parameters of the timer. Recall that these can be set via the GUI, or with register writes:




```
452  /* USER CODE END TIM3_Init 1 */
453  htim3.Instance = TIM3;
454  htim3.Init.Prescaler = 7999;
455  htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
456  htim3.Init.Period = 1000;
457  htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
458  htim3.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
```

The S3 button simply changes the parameters to Timer3. These parameters have been stored in **`main.c`**, in a **`struct`** for convenience and readability.

The respective parameters are then used by the S3_Button ISR to change the operation of the Timer so that the ADC sample timing changes.

Part 1 3 Pts.

Q1 Modify main.c to change the values here.

Calculate, Fill-in, Measure

```
62  const struct adc_sample_config sample_case[] = {
63          {1,     799,    12510}, //First #  in each line
64          {2,     799,    12510}, //Second # in each line
65          {4,     799,    12510}, //Third # in each line i
66          {8,     799,    12510}, // This line is valid,
67          {50,    79,     25029}
68          };
69  /*************** STUDENT EDIT END ******************/
```

Measure your results with the analyzer (ADC_Sample pin, time between samples and total number of samples / cycle). *(I'll run your committed code when grading)*

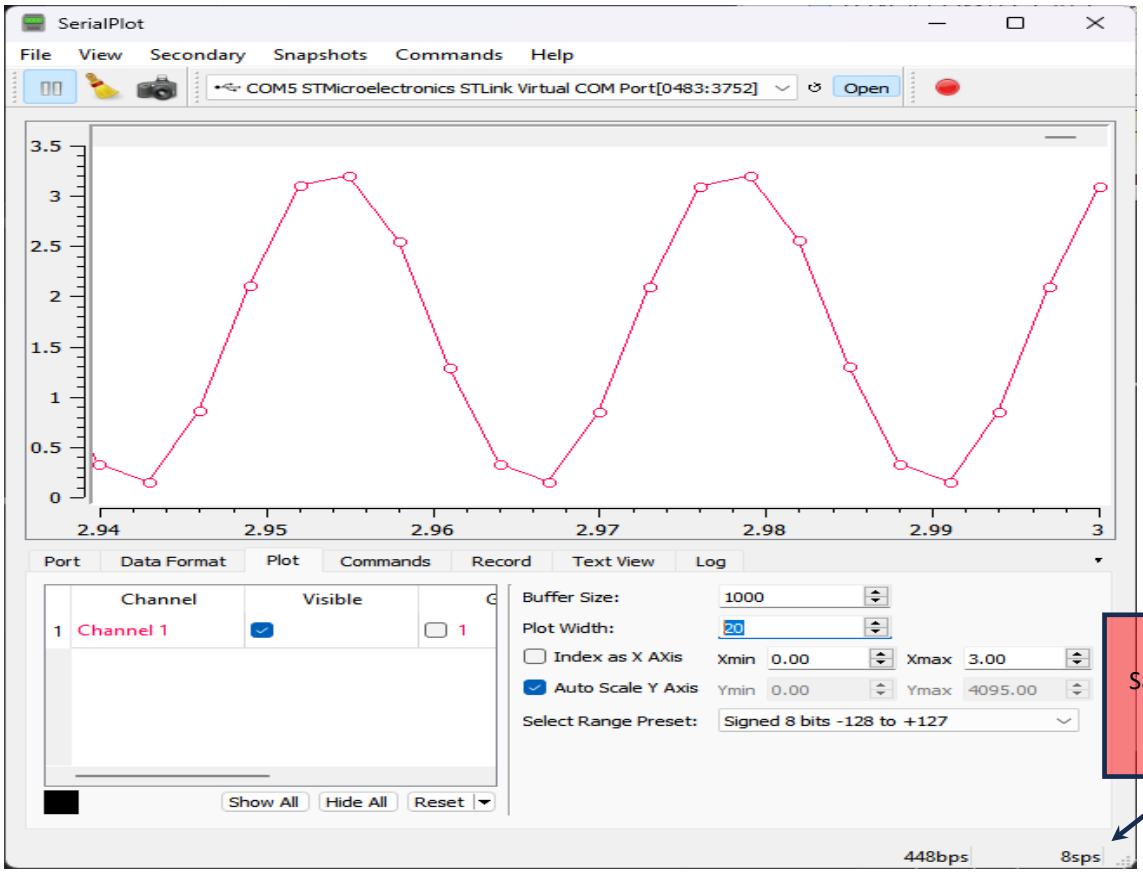
I made the changes and verified the correct sample rates: ☐YES ☒ NO

Part 2: Effect of Sampling Rate

Now that the sample rate is known and selectable, we'll look at the results via the SerialPlotter.

You should see varying accuracies of the re-created waveform. Here's the 8 samples/cycle:

If using the Arduino – note the difference between the output with/without the "Interpolate Switch."



Note the '**sps**' Samples per second. Accurate because our source is 1Hz

Part 2.1 3 Pts.

Q2 Why did the "Period" on the timer need to be slightly adjusted from your calculation? Why didn't the exact number just work?

_____

Q3 In the current approach, could the serial port Baud rate affect the ADC Sample Rate?

_____

Q4 How could you change the code/structure of the ADC sampling, currently controlled by the timer ISR, to make it require less empirical adjustment?

_____

Part 2.2 2 Pts.

Q5 Take/Paste two screenshots from the output of your SerialPlotter screen showing the resulting waveformsfrom the ADC for the two options : one_sample_per_cycle, and two_samples_per_cycle

SCREENSHOT OF **one_sample_per_cycle**

SCREENSHOT OF **two_samples_per_cycle**

Part 2.3 2 Pts.

Q6 What do you see about the **one_sample_per_cycle** plot/results?

_____

Relationship to the Nyquist Sampling Theorem (LINK HERE if you forgot)?

_____

Q7 Practically, is **two_sample_per_cycles** adequate? Could you see aliasing in your **two_sample_per_cycles** plot?

_____

Extra Credit Ideas (Points as indicated) Do something fun/cool!

1.) Button S2 was not available for this lab. Why not? (1 pt)

_____

2.) The serial port was done by using the printf() function. Change it to be a DMA buffer out instead. (5 pts.)

_____

3.) Use another of the timers, set it to fine resolution (µSecs), and trigger it on each start of the sample so you can automatically read how long between samples (5 pts):

_____

4.) Instead of using a timer's ISR to launch a manual S/W-triggered ADC, configure the ADC3 to use the "EXTI Conversion Trigger" instead to automatically (H/W triggered) time the samples. Make any comments about it here (5 pts):

_____