

Lab 5 Report

Requirements:

1. Overview

1.1. Objectives: Why are we doing this project? What is the purpose?

The objectives of this project are to design, build and test a music player. Educationally, students are learning how to interface a DAC, how to design a speaker amplifier, how to store digital music in ROM, and how to perform DAC output in the background. Your goal is to play your favorite song.

1.2. Process: How will the project be developed?

The project will be developed using the TM4C123 board. There will be two or three switches that the operator will use to control the music player. The system will be built on a solderless breadboard and run on the usual USB power. The system may use the on-board switches or off-board switches. A hardware/software interface will be designed that allows software to control the player. There will be at least three hardware/software modules: switch input, DAC output, and the music player. The process will be to design and test each module independently from the other modules. After each module is tested, the system will be built and tested.

1.3. Roles and Responsibilities: Who will do what? Who are the clients?

EE445L students are the engineers and the TA is the client. Students are expected to make minor modifications to this document in order to clarify exactly what they plan to build. Students are allowed to divide responsibilities of the project however they wish, but, at the time of demonstration, both students are expected to understand all aspects of the design.

1.4. Interactions with Existing Systems: How will it fit in?

The system will use the TM4C123 board, a solderless breadboard, and the speaker as shown in Figure 5.1. It will be powered using the USB cable. You may use a +5V power from the lab bench, but please do not power the TPA731 or the speaker with a voltage above +5V.

1.5. Terminology: Define terms used in the document.

Definitions for the terms

SSI – Synchronous Serial Interface is a point to point serial communication system involving a master and a slave unit. The master and slave unit are connected via a clock wire and a data wire, with the master unit setting driving the clock and sending data to the slave unit.

Linearity – how linear the frequency response behavior is for a signal

frequency response – the range of low, mid, and high frequencies a device can attenuate and produce, often characterized by the device's properties to either modify, include, or exclude certain inputs.

Loudness – the power of the sound wave, which is related to the amplitude of the signal.

Pitch – the sensation or quality of the sound as determined by the frequency of the signal

Instrument – device or object used to produce particular notes or sounds

tempo – the rate at which successive notes or sounds are produced

envelope – the decay characteristics of a sound wave

melody – a sequence of musical notes

harmony – the combination of melodies or simultaneously played notes

1.6. Security: How will intellectual property be managed?

The system may include software from StellarisWare and from the book. No software written for this project may be transmitted, viewed, or communicated with any other EE445L student past, present, or future (other than the lab partner of course). It is the responsibility of the team to keep its EE445L lab solutions secure.

2. Function Description

2.1. Functionality: What will the system do precisely?

If the operator presses the play/pause button the music will play or pause. If the operator presses the play/pause button once the music should pause. Hitting the play/pause again causes music to continue. The play/pause button does not restart from the beginning, rather it continues from the position it was paused. If the rewind button is pressed, the music stops and the next play operation will start from the beginning. There is a mode switch that allows the operator to control some aspect of the player.

Possibilities include instrument, envelope or tempo. (Note to students: if you use the internal switches you could rename the switches SW1 and SW2 to match the switches you use) (Note to students: specify exactly what your mode button does.)

There must be a C data structure to hold the music. There must be a music driver that plays songs. The length of the song should be at least 30 seconds and comprise of at least 8 different frequencies. Although you will be playing only one song, the song data itself will be stored in a separate place and be easy to change. The player runs in the background using interrupts. The foreground (main) initializes the player, then executes while(1){} do nothing loop. If you wish to include LCD output, this output should

occur in the foreground. The maximum time to execute one instance of the ISR is xxxx (note to students: replace the xxxx with performance measure of your solution). You will need public functions Rewind, Play and Stop, which perform operations like a cassette tape player. The Play function has an input parameter that defines the song to play. A background thread implemented with output compare will fetch data out of your music structure and send them to the DAC. Again, feel free to change the functional description to match your design.

There must be a C data structure to store the sound waveform or instrument. You are free to design your own format, as long as it uses a formal data structure (i.e., struct). The generated music must sound beautiful utilizing the SNR of the DAC. Although you only have to implement one instrument, it should be easy to change instruments.

2.2. Scope: List the phases and what will be delivered in each phase.

Phase 1 is the preparation; phase 2 is the demonstration; and phase 3 is the lab report. Details can be found in the lab manual.

2.3. Prototypes: How will intermediate progress be demonstrated?

A prototype system running on the TM4C123 board and solderless breadboard will be demonstrated. Progress will be judged by the preparation, demonstration and lab report.

2.4. Performance: Define the measures and describe how they will be determined.

The system will be judged by three qualitative measures. First, the software modules must be easy to understand and well-organized. Second, the system must employ an abstract data structures to hold the sound and the music. There should be a clear and obvious translation from sheet music to the data structure. Backward jumps in the ISR are not allowed. Waiting for SSI output to complete is an acceptable backwards jump. Third, all software will be judged according to style guidelines. Software must follow the style described in Section 3.3 of the book (note to students: you may edit this sentence to define a different style format). There are three quantitative measures. First, the SNR of the DAC output of a sine wave should be measured. Second, the maximum time to run one instance of the ISR will be recorded. Third, you will measure power supply current to run the system. There is no particular need to optimize any of these quantitative measures in this system.

2.5. Usability: Describe the interfaces. Be quantitative if possible.

There will be three switch inputs. The DAC will be interfaced to a 32-ohm speaker. (note to students: you could use 8 ohm speaker)

2.6. Safety: Explain any safety requirements and how they will be measured.

If you are using headphones, please verify the sound it not too loud before placing the phones next to your ears.

3. Deliverables

3.1. Reports: How will the system be described?

A lab report described below is due by the due date listed in the syllabus. This report includes the final requirements document.

3.2. Audits: How will the clients evaluate progress?

The preparation is due at the beginning of the lab period on the date listed in the syllabus.

3.3. Outcomes: What are the deliverables? How do we know when it is done?

There are three deliverables: preparation, demonstration, and report.

Measurements:

The reference voltage is set to 1.25V.

The resolution is voltage change per increment value of our DAC. Using the following equation:

$$(30\text{mV} - (-18\text{mV})) / (105 - 0) = 0.455\text{mV}$$

The resolution of this DAC is 0.455mV

$$\text{Maximum DAC value} = (2 * 1.25\text{V}) / (0.455 \text{ mV/DAC}) = 5483$$

The precision of the DAC is 4096. Using a theoretical DAC value of 1500:

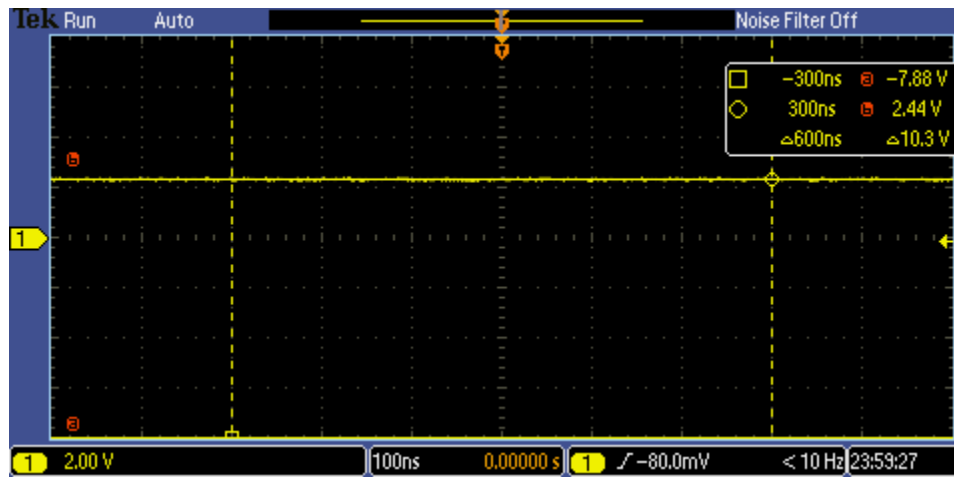
$$(1500 * 2 * 2) / 4096 = 1.465\text{V}$$

The actual voltage for the DAC output of 1500 is 1.83V.

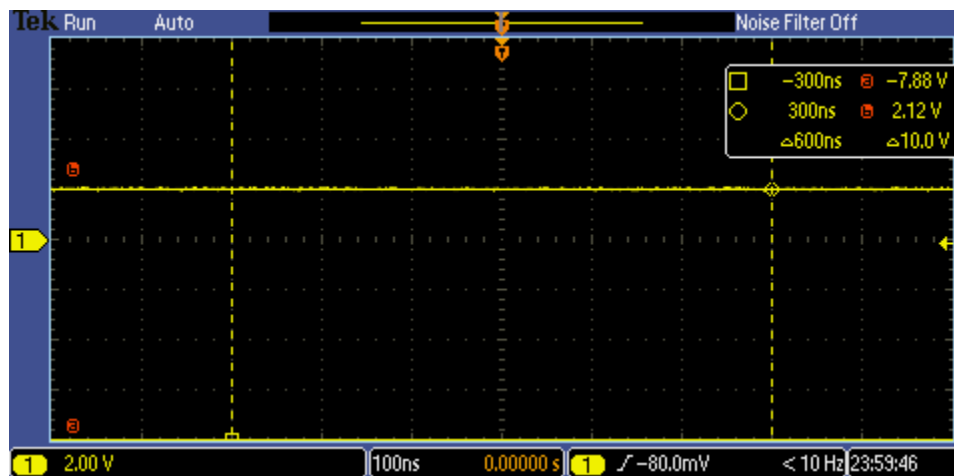
$$\text{Accuracy of the DAC is found using } 1 - (1.83\text{V} - 1.465\text{V}) / 1.465\text{V} = 0.7509 = 75.09\%$$

DAC output measurements:

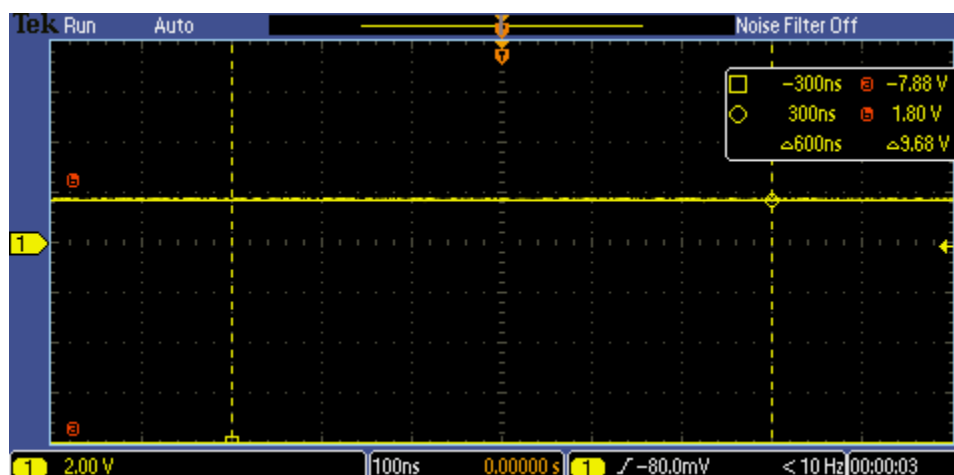
DAC Voltage for 2000 = 2.44V



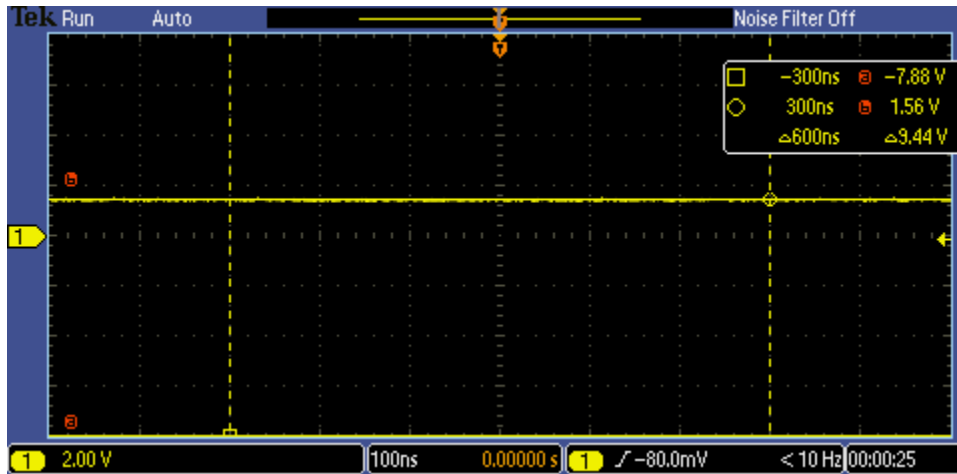
DAC Voltage for 1750 = 2.12V



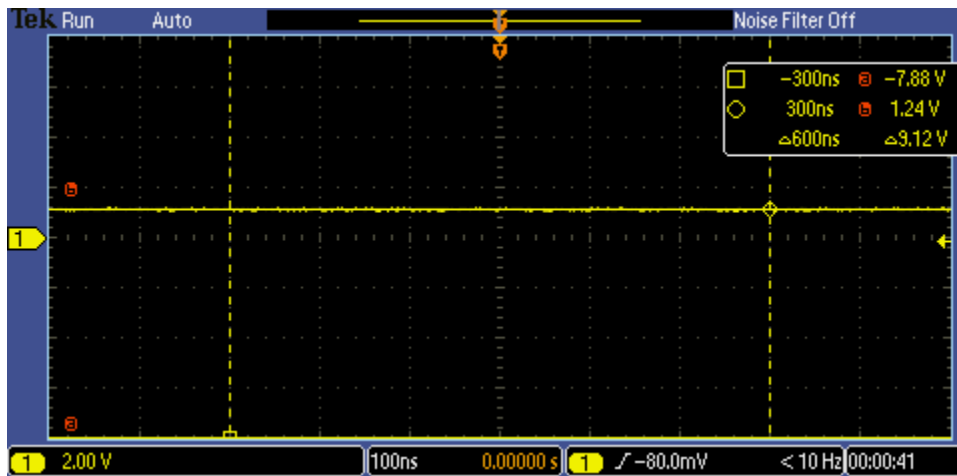
DAC Voltage for 1500 = 1.8V



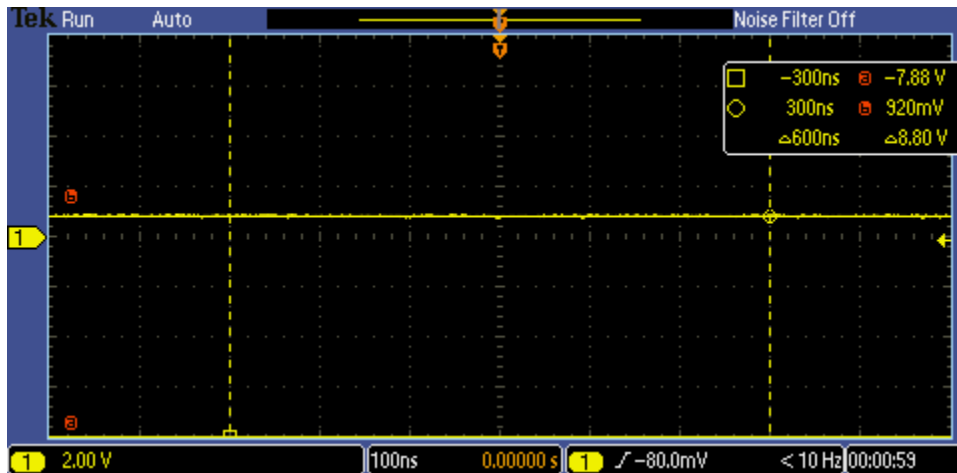
DAC Voltage for 1250 = 1.56V



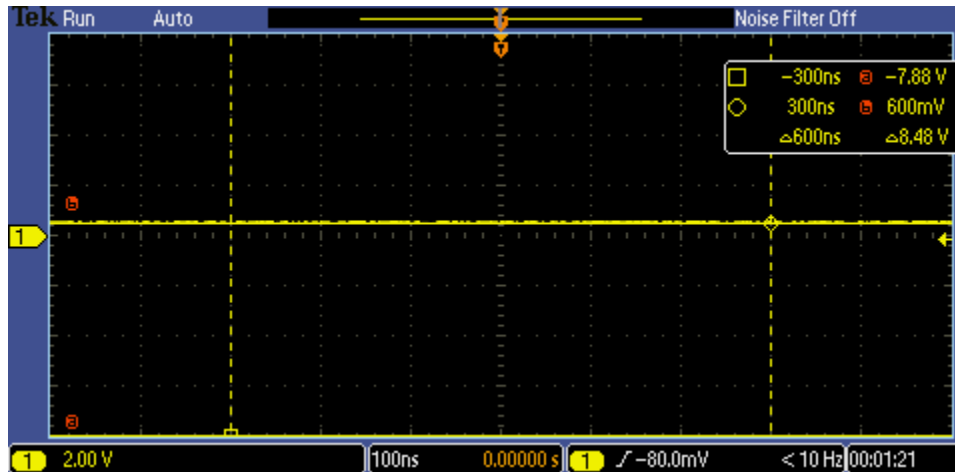
DAC Voltage for 1000 = 1.24V



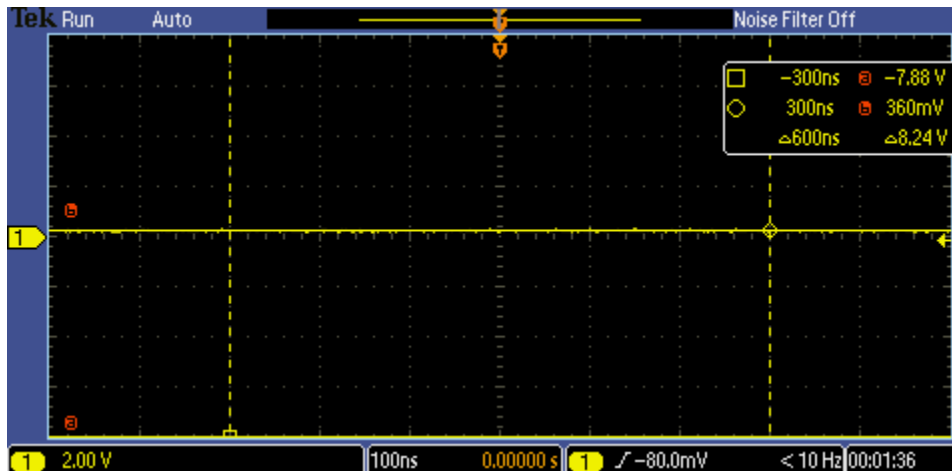
DAC Voltage for 750 = 0.92V



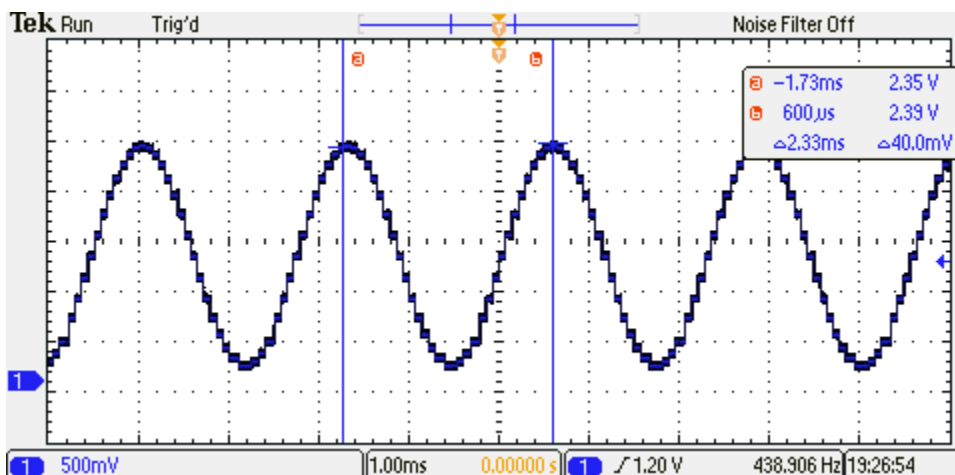
DAC Output Voltage for 500: 0.60V



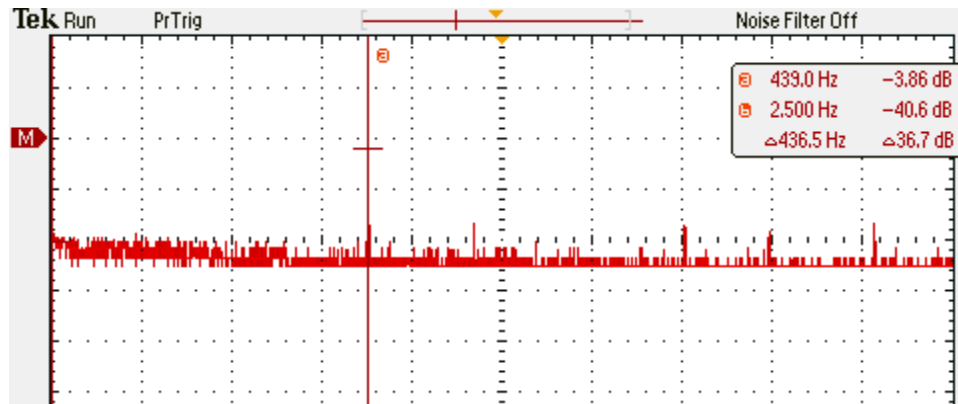
DAC Output Voltage for 250: 0.36V



Time domain frequency at 440Hz (A note):

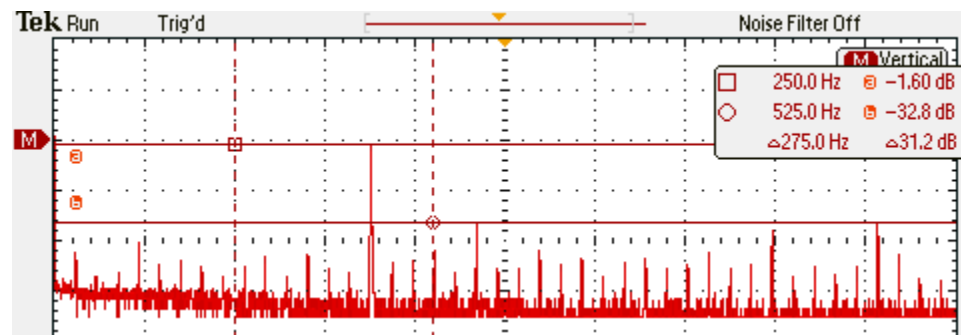


Frequency Domain:



SNR Calculations:

$$\text{signal} - \text{noise} = (-1.6\text{dB}) - (-32.8\text{dB}) = 31.2\text{dB}$$



Analysis:

- Three possible errors on a DAC could include the following:
 - Poor resistor network for the DAC, resistor values not double the previous ladder
 - the operational amplifier could produce noise
 - noise produced with interfering frequencies
- With SPH and SPO set to 0, the interval for which data is available is 25ns. The data required interval depends on both setup time and hold time, so 13ns in total. The clock speed was set to 80MHz resulting in a period of 12.5ns. Since the smallest clock low width is 25ns, the shortest SSI period is 12.5ns + 25ns = 37.5ns. This corresponds to a max SSI clock of 26.6667 MHz. The SSI had a clock speed of 8MHz which is below the maximum.

3. Frequency range of a spectrum analyzer is the sampling rate of the analyzer divided by 2. This is Nyquist Theorem.
4. The speaker was not driven from the DAC because the current required for it to produce an audible volume exceeds the microcontroller's output pin current supply. The opamp (TPA731) resolves the current problem by providing a current gain and a voltage decrease.