Lab 6 grading sheet Circle professor

1) Name Last_____ First_____ EID_____ AC, RY, JV

2) Name Last_____ First_____ EID_____ AC, RY, JV

Use same spelling as listed on Canvas

1. All source files that you have changed or added (like main.c) should be committed to Git.

2. Deliverables 20%:

0) This sheet

Combine the following components in this order into one pdf file and commit it to Lab6 folder in Git before your checkout time. Have this file open on the computer during demonstration.

1) Circuit diagram showing the DAC and any other hardware used in this lab

2) Software Design

Draw pictures of the data structures used to store the sound data

If you organized the system different than Figure 6.6 and 6.7,

then draw its data flow and call graphs

3) A picture of the real scope (not simulation) (part g) like Figures 6.5 or 6.11.

4) Measurement Data

Show the theoretical response of DAC voltage versus digital value (part c, Table 6.3)

Show the experimental response of DAC voltage versus digital value (part c, Table 6.3)

Calculate resolution, range, precision and accuracy

5) Brief, one sentence answers to the following questions

When does the interrupt trigger occur?

In which file is the interrupt vector?

List the steps that occur after trigger occurs and before processor executes handler.

It looks like **`BX LR`** instruction simply moves LR into PC, how does this return?

3. Performance 35%:

Does it handle correctly all situations as specified?

4. Adhere to coding standard 5%:

Good Names have meaning

Variables have units in comments

Consistent indentation

Consistent use of braces

C99 style

	1)	2)
5. **Demonstration 40%**:		

You should be able to demonstrate the four notes. Be prepared to explain how your software works. You should be prepared to discuss alternative approaches and be able to justify your solution. The TA may look at your data and expect you to understand how the data was collected and how DAC works. In particular, you should be able to design a DAC with 5 to 10 bits. What is the range, resolution and precision? You will tell the TA what frequency you are trying to generate, and they may check the accuracy with a frequency meter or scope. TAs may ask you what frequency it is supposed to be, and then ask you to prove it using calculations. Just having three different sounding waves is not enough, you must demonstrate the frequency is proper and it is a sinewave (at least as good as you can get with a 4-bit DAC). You will be asked to attach your DAC output to the scope (part g). Many students come for their checkout with systems that did not operate properly. You may be asked SysTick interrupt and DAC questions. If the desired frequency is f, and there are n samples in the sine wave table, what SysTick interrupt period would you use?

This lab mentions 32 samples per cycle. Increasing the DAC output rate and the number of points in the table is one way of smoothing out the "steps" that in the DAC output waveform. If we double the number of samples from 32 to 64 to 128 and so on, keeping the DAC precision at 4-bit, will we keep getting a corresponding increase in quality of the DAC output waveform?

As you increase the number of bits in the DAC you expect an increase in the quality of the output waveform. If we increase the number of bits in the DAC from 4 to 6 to 8 and so on, keeping the number of points in the table fixed at 32, will we keep getting a corresponding increase in quality of the DAC output waveform?

Extra credit. Option 1: The song should contain at least 5 different pitches and at least 20 notes. There must be at least two separate periodic interrupts: one fast interrupt for DAC output and another slower interrupt to sequence through the notes at the tempo of the song. Although you will be playing only one song, the song data itself will be stored as a data structure in flash ROM, and the device driver will perform all the I/O and interrupts to make it happen. You will need public functions **Song** and **Stop**. The **Song** function has an input parameter that defines the song to play. If you complete option 1, you must show the output of your song on a real scope.

Option 2 extra credit will be useful for Lab 10 and is easier to implement. Find a song represented by a sequence of digital numbers sampled at a fixed rate. Convert the digital numbers to 4-bit or 6-bit integers and output the numbers at that same fixed rate. You need at least 20,000 points stored in flash ROM, you will have to upgrade to the full compiler (there are only 256 kibibytes of flash). If you complete option 2, you must show the output of your song on a real scope. You can only do Option 1 or Option 2, not both

	1)	2)
Total:		