Lab 4 - EE 186: ADCs and DACs

Part 2 - ADC

1. Question 1
    a. You can set the resolution to 6-, 8-, 10-, and 12-bits.
    b. Calculated max sampling rate as F_ADC = 30 MHz divided by number of ADC clock cycles it takes (bits + 3)

Resolution	**Max Sampling Rate**
6 bit	3.3 MHz
8 bit	2.7 MHz
10 bit	2.3 MHz
12 bit	2 MHz

    c. Section 21.4.22 - Programmable resolution (RES) - Fast conversion mode
2. Question 2
    a. A voltage divider is needed so that when the photoresistor has a low resistance, we don't short the circuit, which is what could happen if we directly connect it to the board.
    b. The value of the voltage divider can also be used to set the sensitivity.
3. Schematics in "ee186_lab4_stm32cube_no_default/part2_schematics.png"
4. Demo in "ee186_lab4_stm32cube_no_default/part2_demo.MOV"

Part 3 - DAC

1. Question 1
    a. B=9 -> 12'b01001
    b. V_ref / 2^(12-3) + V_ref / 2^(12)
    c. V_Ref = 3.3V
    d. DAC output voltage = 0.00725 V = 0.00220 V_ref
2. ADC/DAC output voltage difference in "ee186_lab4_stm32cube_no_default/part3_voltage_difference_output.png"
3. Question 2
    a. The sampling rate of the DAC is how often I feed it samples. I set the waveform to have a length of 256 samples. This sets the staircase resolution of the sine wave.
    b. Sine waves only contain one frequency. A square wave contains a large component of one frequency but then smaller odd harmonics. Sawtooth will have even and odd harmonics.

c. We need a capacitor because it can act as a simple high-pass filter, preventing a DC signal but allowing the high frequency notes to go through. Also protects speakers from bass signals.

Part 4 - Musical Instrument

1. Some notes on my implementation
    a. I used a timer interrupt handler to cycle through sine wave outputs. The frequency of the interrupt call is set by the auto-reload register of the timer. This is re-evaluated at the end of each interrupt handler call. This ensures a snappy transition between frequencies because each subsequent call has the most recent ARR. This can be viewed in the main loop by pulling the ARR on a delay interval and printing.
    b. I created a ***blues scale*** from some online frequency tables and mapped the ADC input range onto part of that ***A minor*** scale.
    c. Demo video in “ee186_lab4_part4/part_4_theremin_demo_blues_scale.MOV”
    d. Hard to make out the sound because it was outputting so quietly from my headphones but I tried to hold it up to the mic.