

BBM 434 – Embedded Systems Lab

Lab 7 Instructions

- Analog-to-Digital and Digital-to-Analog Conversion -

Preparation

You will need a Launchpad, a 10K potentiometer, a headphone jack, three 1.5K resistors, three 12K resistors, and a breadboard.

Purpose

In this lab, you will learn to how to configure and use ADCs on your microcontroller. Also, you will generate sound using a DAC over the headphone and build a simple sound machine.

System requirements

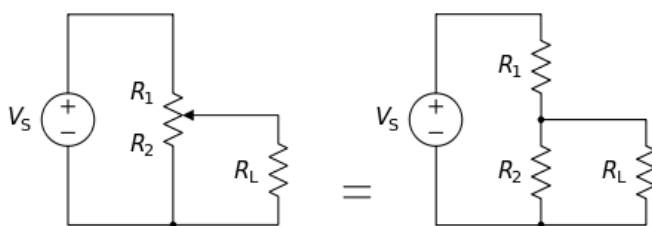
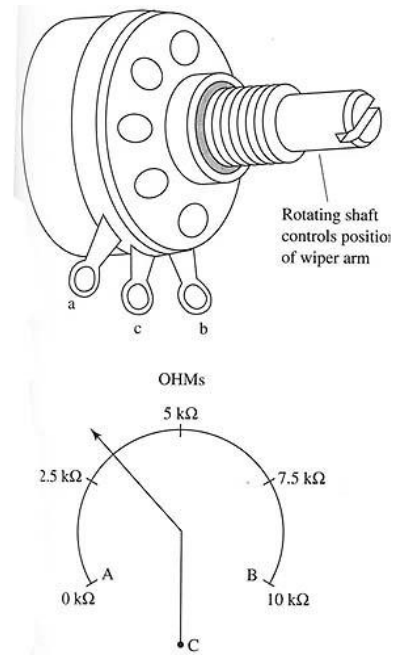
You will develop an annoying sound machine which behaves according to the angle of rotation (rotational travel %) of the potentiometer using an analog voltage reading. According to the voltage reading, you will generate sound using a DAC over the headphones. For example, if the pot is turned 0 degrees, your sound machine should produce one specific note, say Do (C). As the pot turns, the frequency of the sound should change, producing different notes. For instance, if the pot is turned approximately 34 degrees, the output should be the note Re (D).

Note frequencies will be given below for your reference. Moreover, starting from Do is not a hard requirement for this system. You are free to experiment with your implementation. The change in the generated sounds must be obvious though.

Implementation

The potentiometer

Your 10K potentiometer is a variable resistor with three terminals which can be adjusted by the turning of the pot. For example, when the pot is fully turned at 270-degree angle, it will divide the resistors as ($R_1=10\text{K ohm}$ and $R_2=0\text{ K ohm}$). When it is in the middle, it will divide the resistors as ($R_1=5\text{ K ohm}$ and $R_2=5\text{ K ohm}$).



The position of the wiper determines the output voltage of the potentiometer. Your potentiometers may vary. You should consult the datasheet of your potentiometer for detailed specifications.

You will connect Pin A to 3.3V, Pin B to ground and pin C (the pin in the middle) to the ADC port. So, by measuring the voltage in the middle pin, you can deduce the degree of rotation. If it is $3.3\text{V}/2$, for example, you can say that it is turned by an angle of 135 degrees.

ADC software

You can use “C:\Keil_v5\Labware\Lab14_MeasurementOfDistance” as the basis file. Note that if you are measuring angle instead of distance, some comments in that code may not apply.

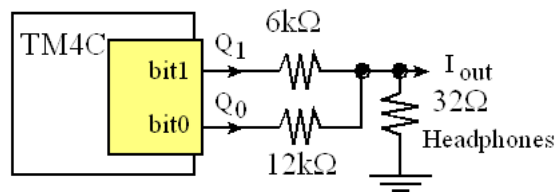
Decide which port pin you will use for the ADC input. Write the ADC software that initializes and samples the ADC. Notice there is an ADC.h file containing the prototypes for the public functions and an ADC.c file containing the implementations. For this part, you are to write **ADC0_Init()** and **ADC0_In**. You can also refer to the code that we discuss in the class.

http://users.ece.utexas.edu/~valvano/Volume1/E-Book/C14_ADCdataAcquisition.htm#Checkpoint14_1

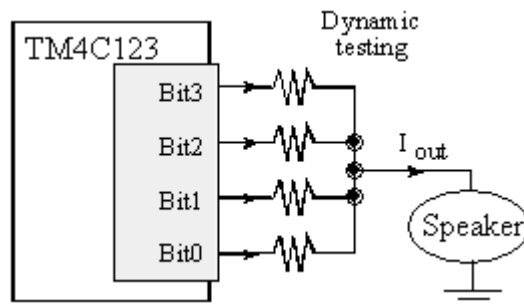
Binary Weighted DAC

The first step is to design and test a 4-bit binary-weighted DAC, which converts 4 bits of digital output from the TM4C123 to an analog signal. You will convert the digital output signals to an analog output using a simple resistor network. If you connect the DAC to headphones, you will be able to hear the sounds created by your software.

A two-bit binary DAC is what we have covered in the lecture:



In this lab you will implement a four bit DAC by using a 1.5K resistor, a 3K resistor (two 1.5K resistors in series), a 6K resistor (two 12K resistors in parallel) and a 12K resistor.



The headphone interface

Build the external circuit on the breadboard. Connect one side of the headphone jack to ground and the other to the DAC. Since the sound is caused by the oscillations, it doesn't matter which side of the headphones go to the microcontroller and which side goes to ground. Figure below shows two stereo jacks. The jack is used to connect the headphones to the circuit. Most jacks have three pins. Luckily for us, sound will be created if we connect any two of these three pins to the circuit, in either direction.



Sound generation

You can use the sine generator we covered in the class as the base code. That code, however, was for a three-bit DAC. Since you will develop a 4-bit DAC in this lab, you need to change the array that represents the sine wave. That code also was to generate a 100 Hz note. In this lab, you will change the frequency to output multiple notes. You can change the frequency by adapting the SysTick reload value.

The following figure shows the notes frequencies.

• Do	• 도	• 262 Hz
• Re	• 레	• 294 Hz
• Mi	• 미	• 330 Hz
• Fa	• 파	• 349 Hz
• Sol (So)	• 솔	• 392 Hz
• La	• 라	• 440 Hz
• Si (Ti)	• 시	• 494 Hz
• Do	• 도	• 523 Hz

For example, you may implement your system so that when the degree of the potentiometer is 0, the sound machine generates the note Do (C), and when the degree changes by about 34 degrees the output becomes the note Re (D), and so on. The notes should continue changing as you slide/rotate the potentiometer.

Find a software like this to check your output tone:

<https://play.google.com/store/apps/details?id=org.cohortor.gstrings&hl=en>

You can check your output with a spectrum analyzer:

<https://play.google.com/store/apps/details?id=com.raspw.SpectrumAnalyze&hl=en>

Pulse Width Modulation for Extra Credit (Optional)

If you wish to get some extra credit in hopes of improving your grade, you need to connect 6 additional LEDs which will behave according to the angle of rotation of the potentiometer. Again using analog voltage reading, you will output the voltage reading to six LEDs. If the pot is turned between 0 and 45 degrees, only the first LED will be on and the others will be off. If the pot is turned between 45 and 90 degrees, the first two LEDs will be on and the others will be off. Moreover, the brightness of the LED will depend on the angle of rotation. For example, if the pot is turned half way, 23 degrees, the brightness of the first LED should be half of its maximum. If the pot is turned 157.5 degrees for example, the first three LEDs should be 100% on and the fourth LED should be at half brightness. The system should react to the turning of the pot very fast.

PWM

For LED with half brightness, you have to feed 1 to the output port half of the time; that is, at 50% duty cycle. You have to adjust the duty cycle depending on your requirements. The duty cycling frequency should be very high so that the LED should not appear blinking.

How to Submit a Written Report

The deadline for the submission is **Friday 03 May 2019 until 13:45** (no further extensions will be allowed!). Submit your work through <http://submit.cs.hacettepe.edu.tr/> (submission via e-mail will not be accepted!) in the following format (**one submission per a group**):

- **b<studentID>.zip**
 - **report.pdf**
 - **projectcodes.zip**