BBM 432 – Embedded Systems
Lab 6
-Analog-to-Digital Conversion and PWM-

Preparation

You will need a 10K potentiometer and 6 LEDs for this lab.

Purpose

In this lab, you will learn to how to configure and use ADCs on your microcontroller. Also, you will use PWM to change the brightness of a LED.

System requirements

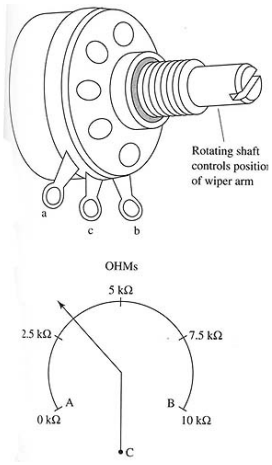
You will develop a software which behaves according to the angle of rotation of the potentiometer using an analog voltage reading. You will output the voltage reading to six LEDs. If the pot is turned between 0 and 45 degrees, only the first LED will be on and the others will be off. If the pot is turned 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.

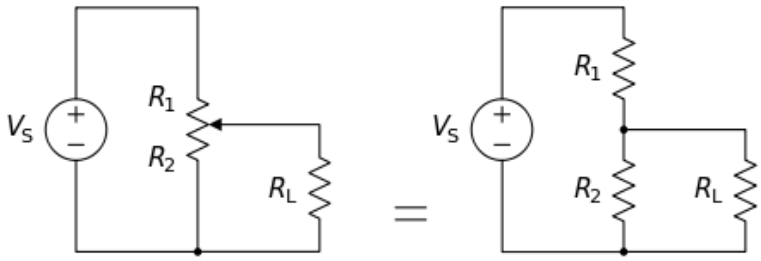
If the pot is turned 157.5 degrees for example, the first three LEDs should be 100% on and the fourth LED should be at half brightness. The system should react the turning of the pot very fast.

Implementation

<u>The potentiometer</u>

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=10K ohm and R_2=0 K ohm). When it is in the middle, it will divide the resistors as (R_1=5 K ohm and R_2=5 K ohm).





The position of the wiper determines the output voltage of the potentiometer. 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 3.3V/2, for example, you can say that it is turned for an angle of 135 degrees.

ADC software

You can use "\Labware\Lab14_MeasurementOfDistance" as the basis file. Note that you will measuring angle instead of distance in this lab. So 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

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.