Lab Exercise #4

rev 2

<u>**Objective**</u>s

- Measure distance with an ultrasound transducer

References

- [1] NXP Kinetis KL25 processor sub-family data sheet (pdf) on class web page.
- [2] NXP Kinetis KL25 processor sub-family reference manual (pdf) on class web page.

Lab 4 Problem Statement

Design, implement, and verify software to setup for and use an ultrasound distance sensor

Design Flow

Keeping in mind the "walk before you run" concept when developing new capabilities, I suggest that the following sequence be followed.

a) Create an *init_dist_TPM0* function to initialize TPM0 and needed GPIO

b) Create a *measure_distance* function that will do the following:

1) Trigger the sensor by setting PortD bit 7 (setting means make it a logic 1)
2) Clear count register

   TPM0->CNT = 0x0000
3) Turn on counter

   TPM0->SC |= TPM_SC_CMOD(1)
4) Clear the capture register flags (just to make sure)

   TPM0->CONTROLS[z].CnSC |= TPM_CnSC_CHF_MASK

   where z = the channel numbers. Do both channels.
5) Delay for 10 us
6) Clear the trigger signal by clearing PortD bit 7
7) Poll the CHF flag of TPM0 channel 0 until it is set

   TPM0->CONTORLS[0].CnSC & TPM_CnSC_CHF_MASK
8) Poll the CHF flag of TPM0 channel 1 until it is set
9) Read the two capture registers registers

   TPM0->CONTROLS[0].CnV

   TPM0->CONTROLS[1].CnV
10) Calculate distance in millimeters (if clock is 1.5 MHz divide by 9. This is close)
11) Return value to caller as an uint32_t

(Continued next page)c)

Create a main.c file that does the following:

- Does initialization
- Has an endless loop:
    - call measure_distance
    - display the returned distance in millimeters on a virtual terminal using UART2.

To Turn In

- In the “comment header” of your main.c file report success, failure, or other observations
- Submit your main.c file to a D2L drop box
- Zip up your complete lab 4 uVision project and submit to the D2L drop box