Chris Bird

Colin Lee

EEC 172 – Embedded Systems Design

Due 17th March, 2015

Lab 5 Report – ADC/DAC Module and uDMA Connectivity

In this lab, we were to use the internal ADC module from the TivaWare microprocessor to read in a signal produced by a function generator, convert it into a digital signal and output it to a MAX549 DAC. Using an internal timer set to 10 kHz, we trigger an ADC interrupt every 1/10,000 of a second to read from the function generator. Inside this handler, we fill up an array of 128 length and use that data to calculate an average and ones complement of the average. When this signal is sent over SSI to the DAC, it gets put onto two different channels, OUTA and OUTB. We send the normal data to OUTA, and the ones complement (inverse of the signal) to OUTB. Both of these outputs are connected to an oscilloscope, which displays the analog output of the DAC.

In our code, the only thing done in the ADC interrupt handler is the storing of data. The rest of the calculations are done in our main while loop, which is controlled by a switch on the processor. When this switch is pressed, it enters the calculation loop where it prints out the entire contents of the buffer (which can be seen attached to the report) along with the calculated average and ones complement.

PART TWO GOES HERE

For the third part of the lab, instead of using uDMA to read the data and analyze it, we were instructed to use a ping-pong buffer. This ping pong buffer allows for faster realization of answers since there will be two separate arrays being accessed at once. The logic behind this is that while the ping buffer is being filled, the pong buffer is being read and analyzed and vise-versa. It also allows the data that we are working on to remain constant throughout the calculations rather than the values being changed while we are using them. This makes for atomic arithmetic and more accurate results. The two things we needed to analyze from our data was the waveform, such as square or sin wave, along with the frequency.

To calculate the frequency, we checked through the inactive ping-pong buffer and looked for the lowest value, once this was obtained, we then looked for the next one, and counted how many samples were between them. This number was then multiplied by two to give the total number of samples in one period, and then further changed with our sampling rate to get our final period time. As the final step, we converted the period to frequency using simple division, and output our answer.

To determine the waveform type, while we were calculating the frequency, we looked for the number of values near each of the high values, this number was then split into ranges which helped determine the type of wave. If there were many near the peak, it was a square wave. If there were almost none, it was a triangle wave, and if it were in the middle, it would be sin. However, this is not accurate enough for triangle vs sin, so we checked the slope after the next couple points, and compared these slopes. If they were very similar it was definitely triangle, otherwise it was a sin wave.