Lab7: DMA Driven DAC Waveform Generator

OBJECTIVES
Design and implement an efficient waveform generator on the XMEGA platform using DMA (Direct Memory Access) and on-board DAC (Digital to Analog Converter).

REQUIRED MATERIALS
- EEL 3744 (uPAD and uPAD Proto Base) Board kit and tools
- Digilent Analog Discovery (DAD) kit
- XMEGA documents
  - doc8331, doc8385, doc0856

YOU WILL NOT BE ALLOWED INTO YOUR LAB SECTION WITHOUT THE REQUIRED PRE-LAB.

PRELAB REQUIREMENTS

REMEMBER:
You must adhere to the Lab Rules and Policies document for every lab.

All code must be written in C. Any part of this code written in assembly only will result in loss of points.

This lab will be more specification driven and will not explain exactly how to do everything. It is VERY important that you read the ENTIRE lab handout, and the DMA and DAC sections of the XMEGA manual (doc8331) before starting this Lab.

PART A: Digital-to-Analog Converter
In Lab 5 you used the XMEGA Analog-to-digital converter to input analog signals and convert them to their equivalent 8- or 12-bit digital representations. For this lab, you will do the exact opposite. In order to generate a waveform, sine for example, the output must be able to output many points between the high and low voltage of the system. This means the output must be an analog signal. For this, you will use the XMEGA on-board Digital-to-Analog converter. Similar to the ADC, the DAC takes in a 12-bit unsigned value and outputs the appropriate analog voltage. Before continuing, be sure to read section 29 of doc8331.

PART A SPECIFICATIONS:
For this part of lab, simply use either DAC channel (CH0 or CH1) on either available DAC equipped ports. Use AREFB (as you did in the ADC lab) for a more accurate reference. You will include your initializations in your lab document. To verify functionality, use your DAD board in the appropriate analog measuring mode to measure your DAC outputting 1.7V.

PRELAB QUESTIONS:

1. If the DAD board were not available, nor any conventional desktop measuring tools, how could you verify that your DAC were functioning properly?

PART B: Timer/Counter Driven Waveform
Now that you can output an analog signal, you can turn that into a wave form. The objective of this part of the lab will be to output a simple sine wave using a look-up table and a Timer/Counter.

PART B SPECIFICATIONS:
Using your DAC configured to the specifications described in Part A of the lab, generate a sine wave with at least 64 data points per period, peaks between 0V and AREFB, and with a frequency of 100 Hz ±2%. As before, this code and the screen shots verifying functionality will be included in your lab report.

While it is not very difficult to use a program such as MATLAB to build your look-up table, it is not required that you generate your own. It is suggested that you find a calculator on-line to build one for you, like the one at http://www.daycounter.com/Calculators/Sine-Generator-Calculator.phtml (for generating sine waves).

PRE-LAB QUESTIONS:

2. You probably noticed that your sine wave does not look very good. While at least meeting specifications, how could you increase the quality of your sine wave?

PART C: DMA Driven Waveform
Using a Timer/Counter is the best way to get a precision waveform, but as you probably found out in Part B it can be a little tricky using polling or interrupts. Every time the processor has to execute code, it adds time between when you wanted something to happen and when it really happened. An easier way to generate the same sine wave would be to use Direct Memory Access (DMA) to move data from one point to another without processor intervention.

When configuring DMA, there are a few things that need to be considered.
- How many bytes will be transferred in a burst (one piece of data)?
- How many bytes are in the complete transfer (maybe there is more than one burst)? Is it just a single burst?
- How many times do you want to move this data? Do you want to move it forever?
- Where should the DMA get the data from? Will it take more than one byte to get all of the data? If so, from where the first byte was, where should DMA look next?
Lab7: DMA Driven DAC Waveform Generator

- Where should the DMA put the data? Again, if there is more than one byte in the destination, where should it put the next?
- With the source and destination locations, does DMA need to point back to their original locations after a transfer?
- What is going to trigger this transfer? ADC complete? An interrupt?

As you can see, there can be a lot of settings to consider with DMA. It is advised at this time to take a moment to look over section 5 of doc8331. Do not wait for the last minute to configure DMA for the first time!

PART C SPECIFICATIONS:
Follow the same specifications as Part B of this lab, i.e., generate a 100 Hz (±2%) sine wave using at least 64 data points per period. Include screen shots of the Oscilloscope confirming the functionality of Part C.

PRE-LAB QUESTIONS:
3. How many DMA channels are available on the XMEGA?
4. How many different options are there to trigger the DMA?

PART D: Advanced Waveform Generator
At this point the hard work is done, but your waveform generator is not very useful. For this part of the lab you will use your 4x4 keypad you have used in several previous labs. Make sure you are able to easily report the pressed key because this is what will determine what you output. If your keypad code needs work, it is suggested you do this now.

Just like the sine wave before, you will generate another look-up table of at least 64 data points per period. There can be more data points if desired, but it is highly suggested that both of your tables are the same size. The new table will be of a triangle wave. Integrate this new wave into your code the same way you did the sine wave. (A triangle wave calculator is available at the following URL: http://www.daycounter.com/Calculators/Triangle-Wave-Generator-Calculator.phtml.)

The idea of this part of the lab will be to control which wave to output and at what frequency. By pressing the asterisk (*) your waveform generator should output a sine wave. Pressing the octothorpe (#) should change the wave to the triangle wave. Each other key should be a multiplier for your 100 Hz standard frequency. This means that by pressing 0 your wave should turn off (0 Hz), 1 should output 100 Hz, 2 should output 200 Hz, A (decimal 10) should output 1000 Hz, B (decimal 11) should output 1100 Hz and so on. Every key should serve some sort of functionality. For simplicity, the initial output should be the sine wave turned off.

PART D SPECIFICATIONS:
Your advanced waveform generator designed in Part D must use DMA to supply the on-board XMEGA DAC with the data points. The sine wave and the triangle wave must consist of at least 64 data point, although you may use more. The frequency of either wave may have no more than ±2% error measured in Hz. Each of the 14 hex numbered keys 0 through D) should act as a multiplier of the base 100 Hz frequency (i.e., 100 Hz × 0 = 0, 100 Hz × 1 = 100 Hz, ..., 100 Hz × 0xA = 1000 Hz, 100 Hz × 0xB = 1100 Hz, etc.). The only required screen shot is of the triangle wave at 500 Hz. Please include this single screen shot for this section in your lab document.

PRE-LAB QUESTIONS:
5. While you were not asked to implement this on your board, explain how one might vary the amplitude of an output wave, much like the frequency was varied.

IN-LAB REQUIREMENTS
As per the lab rules and policies, all parts of the lab must be finished and submitted on Canvas BEFORE your designated lab period. Failure to do so will result in a 0 for the incomplete portions of the lab.

You will submit your pre-lab report with all (5) pre-lab questions and screen shots included. Your source code should also be included in your lab document and submitted as an ASM. Lab documents will only be accepted in PDF form.

You will demo parts B, C, and D in lab:

You will be expected to supply live evidence that your design is working within the specifications provided. It is advisable that you bring your DAD board to lab for this purpose.