ELEC 444 – Advanced Digital Design
Spring 2014

Lab 4: Digital Audio

Due 2/27/14 at 8:00 am

Introduction

This lab provides further experience interfacing with the Zynq7 ARM9 core and using the integrated hardware features. You will create a program which generates audio to drive a standard pair of speakers.

Task

Create a system to play music on a set of speakers using the ARM core of the Zynq7, a PModAMP1 audio amplifier, and a standard set of speakers. The program should read a list of notes specifying the period and duration of each note. It should generate a sequence of square waves with the corresponding periods and durations. Use two timers to accurately create the period and duration for each note. A CSV file containing the period and duration (in µs) for each note in the test song Fur Elise is provided.

Minimum Requirements

The following are the minimum requirements for your system. You may of course add additional functionality.

- The system must output stereo audio that plays the provided portion of Fur Elise.
    - The period and duration of each note may be hardcoded as an array in your program.
    - Each channel can output the same notes.
- The system should use two timers, one to count the period and one to count the duration.
- Use the oscilloscope to verify the output and take a snapshot for your submission.

Suggestions

- Use a prescalar of 200 with a timer running at half the core frequency for the period timer. This will result in a counter that counts at 0.6 µs. Use this counter to count to *half* the period of the note and then toggle the output
- Select a large prescalar, like 256, for the duration timer.
- The private core timers can be accessed either through the available library functions or through direct control register access. The global timer does not appear to have a set of library functions and so should be accessed by reading/writing the control/data registers.
- Convert the times listed in the provided Fur Elise CSV file into counts for each of the timers for each note. Include these counts as an array (or arrays) you access in your program. When converting the provided times to counts you can round off any fractional portion of a count.

Background

Timers

The Zynq7 ARM cores include three types of timers, a per-core private 32-bit timer, a global 64-bit timer and two shared 16-bit triple timer counters. Each has slightly different abilities and features.

Private 32-Bit Timer

The private timers count at a frequency one half the core frequency (667 MHz/2 for the ZedBoard). The counters count down from the value loaded into their count register. The input clock can be prescaled by any value between 1 and 256 inclusive.

Global 64-bit Timer

The global timer counts *up* at a frequency one half the core frequency. The input clock can be prescaled by any value between 1 and 256 inclusive. The counter counts up from zero. The upper and lower half of the counter must be read separately. Correctly obtaining the current count value requires three loads. First the upper half of the count is read, then the lower half, and then again the upper half. If the two upper half values match then the count value is valid.

Triple Timer Counters

Each triple timer counter provides three separate 16-bit counters/timers. The timers can run off of a number of different clock inputs and the input clock can be prescaled by powers of two from 2^1 to 2^{16}. The timers have a number of other features, including the ability to automatically toggle an output based on the current count relative to a threshold value. This can be used to create a variety of pulse width modulated outputs automatically. Unfortunately for this lab, none of the output pins of the triple timer counters connect to the MIO pins connected to the JE PMod connector.

PModAMP1

The PModAMP1 amplifies low power audio signals and provides both monophonic and stereo output via $1/8^{th}$ inch audio jacks. The module includes a potentiometer for volume control and a low pass filter to turn a digital pulse width modulated signal into an analog voltage. The ARM core can communicate directly with the PModAMP1 via the JE PMod connector and MIO outputs. The mapping between the JE PMod pins and MIO ports is shown in Table 1.

	Zynq7 Pin	MIO Pin		Zynq7 Pin	MIO Pin
JE1	A6	13	JE7	G6	0*
JE2	G7	10	JE8	C4	9
JE3	B4	11	JE9	B6	14
JE4	C5	12	JE10	E6	15

Table 1. JE PMod Pin Mapping

*The ZedBoard HW User's Guide states that this should be MIO7, but package information says that pin G6 is connected to MIO0

Submission

You must submit a copy of the following items via Moodle by the indicated due date. The same guidelines as applied to the first lab apply to this lab. Submit the items in a single document, making sure code is presented in a monospaced font.

1. Paragraph discussing how your approach the problem and structured your code.
2. C Code of your working program. Code should be well commented and neatly structured, including good use of functions.
3. Capture of the oscilloscope output measurement.
4. The amount of time you spent on this lab and any suggestions, comments, or complaints you have about the lab. This will not impact your grade, but will help to improve labs in the future.

Extra Credit

Compose and play a different tune that, at least for some of the song, plays different notes on the right and left channel.

As some structure and background, the A above middle C (called A4) is 440 Hz. An octave spans a factor of 2 in frequency. There are twelve notes in an octave spaced evenly on a geometric scale, so each is separated in frequency by a factor of $2^{(1/12)}$.

Note	Frequency (Hz)
A3	220
A sharp / B flat	233.1
B3	246.9
C3 (middle C)	261.6
C sharp / D flat	277.2
D3	293.7
D sharp / E flat	311.1
E3	329.6
F3	349.2
F sharp / G flat	370.0
G3	392.0
G sharp / A flat	415.3
A4	440
A sharp / B flat	466.2
B4	493.9
C4	523.3
C sharp / D flat	554.4
D4	587.3
D sharp / E flat	622.2
E4	659.2
F4	698.4

F sharp / G flat	740.9
G4	784.0
G sharp / A flat	830.6
A5	880

The duration depends on an arbitrary choice of tempo (speed at which the piece is played). If a whole note is chosen to be 1 second long, other notes follow accordingly:

Duration	**Seconds**
Sixteenth	0.0625
Eighth	0.125
Quarter	0.25
Half	0.5
Whole	1