PROCEDURE

Assume the resistance of your headphones is 32 Ω. Power delivered to the headphones is *V***I* . Assume the output high voltage on PA2 is 3.3V. Let *R* be the series resistor, shown as 1500 Ω in Figure 12.1. Using Ohm's Law we can estimate the current delivered to your headphones

$I = 3.3V/(R+32\Omega)$

The power delivered to the headphones will be

$P = I^2 * 32\Omega = (3.3V/(R+32\Omega))^2 * 32\Omega$

If R=1500Ω, P will be 0.15 mW. You should limit the power to the headphones to less than 1 mW.

Sound will be created only when the voltage across the speaker oscillates. The frequency of oscillation will determine the pitch of the sound. A frequency of 440 Hz will be the note A, which is above middle C on the piano. To output silence, it requires less energy to leave the output pin low. An unchanging high output results in silence from the speaker, but it also causes current to flow through the electromagnet in the speaker for no useful purpose.

Part a) First design and test the system in the simulator. Just like most of the other labs, the **TExaS_Init** will configure the PLL to run at 80 MHz. You can observe the two Port A pins using the logic analyzer available in the simulator. Figure 12.3 shows a zoomed-out view illustrating the proper relation between input and output. Figure 12.4 shows a zoomed-in view illustrating how to measure the frequency of the tone. In previous labs we created delays with software loops. Software delays are difficult to get the time just right and are extremely inaccurate. With the SysTick interrupt you will find creating the 440-Hz wave with an 880-Hz interrupt will be both straight-forward and accurate. Furthermore, the same software will be accurate both in simulation and on the real board.

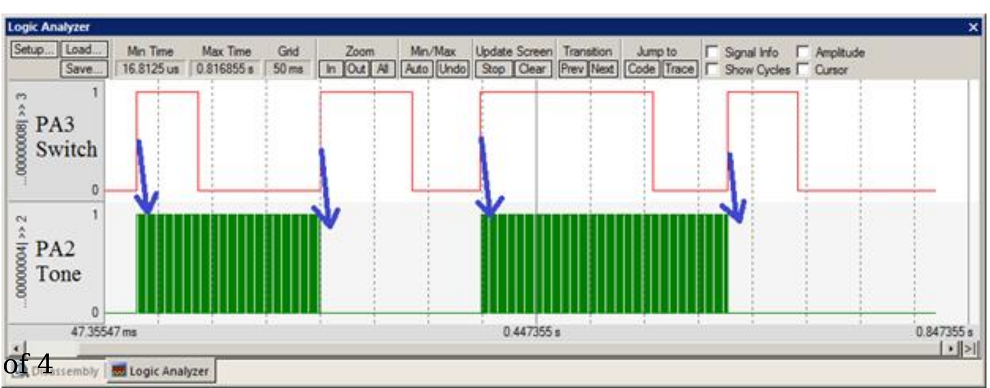

Figure 12.3. Logic analyzer waveforms showing Lab 12 running in simulation mode.

We expect you to use a SysTick ISR and look at the switch during this ISR. You will need two global variables. One global variable to remember the previous switch value at the last ISR execution (pressed or not pressed), and one global variable to know what to do (toggle or quiet). If the switch was not pressed during the last ISR execution and is pressed during the current ISR, then you know the switch was just pressed.

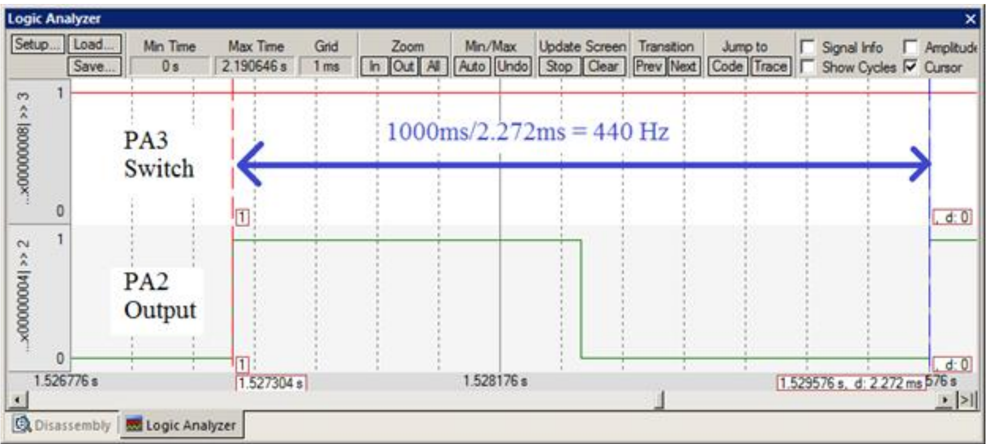

Figure 12.4. Logic analyzer waveforms showing Lab 12 running in simulation mode zoomed in to see the tone is exactly 440 Hz.

GRADING IN SIMULATION MODE

0:00 / 1:53 1.0x

PROFESSOR JONATHAN VALVANO: Let's show you

how to get your grade in Lab 12 Simulation Mode.

We begin in edX, and take the four digit number from the screen

here, this one right here, Copy.

We go over to Keil.

And since we're going to be in Simulation mode,

we'll check the options to make sure that we

are set up to do the simulation in Lab 12.

We will compile our program, which is to build, and then we will download it and start the debugger in Simulation mode,

not on the board.

You can see I've set up the Logic Analyzer to measure both the input

and output pins during grading.

Help

And down here, I will paste in the number I got from edX, Paste,

and now I'll hit the Grade button.

And we can watch what it's doing by looking at this window here,

the Command window.

And I'll zoom out, so you can see all of what's going on.

And you can see during initialization, it checked the ports.

Then it put the switch high, and the toggling started.

Making the switch go low did not turn it

There are six steps to the simulation grader:

0) Initialization tests will look specifically to make sure SysTick is interrupting at 880Hz (in particular there is only one value the grader will accept for the SysTick RELOAD register);

1) It will turn the switch off and make sure the output is low;

2) It will turn the switch on and make sure the output toggles;

3) It will turn the switch off and make sure the output continues to toggle;

4) It will turn the switch on and the output should go low;

5) It will turn the switch off and the output should remain low.