Education Objective
The educational objective of this demo is to become familiar with IP cores used to interface with the audio CODEC provided on the DE1-SoC board.

Technical Objective
The technical objective of this laboratory is to design an embedded system for the Nios II processor and DE1-SoC that will load an audio (.wav) file into the external SDRAM and play it through the audio CODEC and an attached speaker.

Demonstration Procedure

1. Download the **Audio_Demo_Support_Files** from MyCourses and unzip the folder.

2. Create a folder named **Audio_Demo** and within it create a folder named **Audio_Demo_quartus**.

3. Create a new Quartus project and name it **Audio_Demo**. Put the **nios_system.qsys** file and the **Audio_Demo.vhd** file in the **Audio_Demo_quartus** folder.

4. Open QYSYS and load the **nios_system.qsys** file. This system has many of the standard components. New for this demo are:
    a. The **audio_video_config_component** – this is a core that communicates with the CODEC via the I^2C bus. It initializes the CODEC when the design is programmed into the FPGA.
    b. **The audio component** – this core manages the reads from CODEC (for microphone data) and the writes to the CODEC (for speaker data)

    Additionally there is an **SDRAM controller** for interfacing with the off-FPGA SDRAM. We will use that for storing the audio data to be played. Because the SDRAM is being used, the **clock source core** is used to provide the *SDRAM clock at a 3 ns offset* from the 50MHz clock. The onchip memory is used for the source code storage (you can tell because it is connected to the instruction master). Finally, an **interval time** (timer_0) is included as a periodic interrupt generator. It has been configured to fire an interrupt every 20.48 us (1/sample_rate).

5. Generate the QSYS. There will be 1 warning, but it is okay to ignore. Do not connect the interrupt from the audio core.

6. In Quartus add the **nios_system.qip** file and the **Audio_Demo.vhd** file to the project. Import the pin assignments. Compile and program the board.

7. Open NIOS II Software Build Tools for Eclipse. Be sure to point your workspace to the Audio_Demo_quartus/software folder. Create a new application and BSP from template.

Call the app **Audio_Demo_App**. Be sure to select the correct nios_system .sopcinfo file from the Audio_Demo_quartus folder.

8. Move the **Audio_Demo.c** and the .wav file to the application folder. Copy the system.h file from the BSP to the App.

9. Open the BST editor (right click on Audio_Demo_App_bsp > NIOS II > bsp editor). Click on the **"Software Packages"** tab and click to enable **alter_hostfs**. This will allow you to read a file from the computer via the C program.

10. Click on the **"Linker Script"** tab and left click on each reference to **new_sdram_controller_0** in **Linker region names** area and change them to onchip_memory. This will ensure that all of the code exists in onchip memory so it does not get overwritten when the audio data is loaded into SRAM.

11. Generate the BSP, copy the **system.h** file and build the project.

12. Right click on the application and choose Debug As > NIOS II hardware

13. In the debug perspective, click on Resume. The program will go away for a LONG time. It is loading the file from the file system to the SDRAM. Be patient.

14. You will know when it is done loading when you see the message "file read". You should also hear audio at this point.

15. Be sure to edit the C program to change the FIRST_TIME def from 1 to 0. If you don't do this, the file will load every time you try to run the program. The data will remain in the SDRAM as long as the board remains on.

16. Look in the **Audio_Demo.vhd** file and you will see where I have provided a mechanism to view the CODEC signals on GPIO pins. View the audio sample clock and data on the oscilloscope. Review the C code to understand how the samples are stored and retrieved from SDRAM.

17. Demonstrate your working system for credit.
18. Save everything as we will be coming back to this next week.