Third Prize

Driver Assistance Tool

Institution: **Hanyang University**

Participants: **Jonguk Song, Hwanjun Kang, Kangsub Kwack**

Design Introduction

This driver assistance tool detects the presence of a human voice and adjusts the volume of a car's audio system accordingly. Car audio system manufacturers are the primary customers for such a device.

Our project implements a complete product that can play **.wav** files and adjust to the system's environment (in this case, the presence of a human voice). The design has two parts: an MP3 player and a voice-activated volume adjustment mechanism. Each part uses a Nios® II processor.

Function Description

When a human voice is detected, the system pauses playback of the MP3 audio file and then resumes playing. The volume is adjusted as the music streams. To perform this function, the design contains:

- A Nios II processor (named CPU0) that plays the MP3 file and controls the system delay and the attenuation coefficient of the car's audio system.
- A secure digital (SD) memory card that contains the MP3 files to be played.
- An audio codec that interfaces with the speaker that plays the MP3 files and the microphone that detects the driver's voice.
- A fast Fourier transform (FFT) block that transforms the audio signal into its real and imaginary frequency components.
- A second Nios II processor (named CPU1) that monitors the magnitude of the driver's voice and instructs the audio codec function to reduce the volume.

The system's operation is described as follows:

1 One data frame (512 bytes) is read from the SD memory card and is saved into a digital-to-analog converter (DAC) buffer.

2. The data, which eventually becomes the MP3 sound, is multiplied by an attenuation coefficient, thereby setting the volume. The result is saved to a delay buffer.

3. The CPU0 processor delays the raw data by sending a delay value to the delay buffer.

4. The audio codec converts the human voice data received from the microphone and digitizes the audio using an analog-to-digital converter (ADC). The delayed data is then subtracted from the ADC data.

5. The resulting data stream feeds into the FFT block, which transforms the data into its frequency components and stores the real and imaginary components into the Real_Result_FIFO and Imaginary_Result_FIFO buffers, respectively.

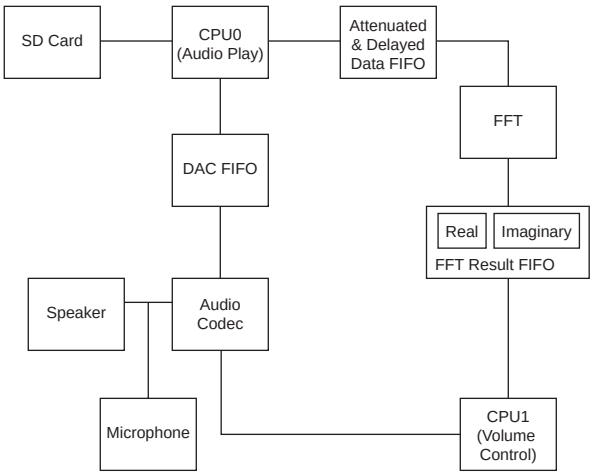
6. The CPU1 processor monitiors the magnitude of the FFT result, which is filtered to respond to the audio range of the human voice.

7. If the magnitude of the FFT result (i.e., the volume of the driver's voice), is greater than a preset trigger level, the CPU1 processor commands the audio codec to reduce the volume and tells the CPU0 processor to set another attenuation coefficient and delay.

Design Architecture

Figure 1 shows our project's design architecture. The **.wav** file to be played is saved into a buffer and the microphone receives a **.wav** file of a human voice. The system subtracts the **.wav** files with delay and gain and performs an FFT on the result. Next, the system measures the magnitude of the voice frequency region and sets the music volume for the received voice level.

Figure 1. Design Architecture



Design Description

We built our design using the following general steps:

1 Using SOPC Builder, we created a DAC FIFO buffer to play the MP3 files. Additionally, we created attenuated and delayed data FIFO buffers to subtract the recorded data from the original data. The design sends the subtracted data to the FFT block.

2. We designed an I^2C control module as an audio codec that is controlled by a Nios II processor. We can control the audio codec with software written using the Nios II Integrated Development Environment (IDE).

3. We incorporated the FFT MegaCore® function into our design.

4. We linked all of the modules and assigned addresses and interrupt requests (IRQs) to each module.

Design Features

Our design consists of two Nios II processors and several intellectual property (IP) functions.

- One processor has an MPEG audio decoder (MAD) application, making it function as an MP3 player.
- The other processor communicates with the correlator and FFT blocks, and determines whether the driver is speaking. This processor also controls the MP3 player's volume.
- The correlation block compares input sounds with MP3 sounds, and computes the delay between the input and MP3 sounds.
- The FFT block analyzes the input sounds and decides whether the sounds are human.

We used system-on-a-programmable-chip (SOPC) concepts to implement the design. SOPC Builder helped us to select IP blocks easily. The Quartus® II Compiler automatically recognized the IP blocks selected for the design and generated the required IP block device drivers and added them to the system library that the application software uses.

Conclusion

This project was a good experience for us. We learned a variety of things during this project, including:

- We learned how to design hardware with the Development and Education (DE2) board.
- We learned how to use the Quartus II software, SOPC Builder, and Nios II IDE. We think they are very useful for creating digital architectures.
- We learned how to work on a team project, including how to cooperate, how to divide tasks among members of the team, and how to create a schedule.