Adaptive Noise Cancellation

Group 8

Gur Kohli
gkohli@sfu.ca
301215504

Svetlana Borkovkina
sborkovk@sfu.ca
301215298

ENSC 452 - Advanced Digital System Design
April 6, 2018

Introduction

- Adaptive noise cancellation filters cancel out ambient noise from a noisy source by using FIR filters with adaptable weights
- Ambient noise is independently recorded and provided as noise input to filters
- Weights are updated to achieve Least Mean Square (LMS) error between noise signal and FIR output
- Used in high end noise cancellation headphones, which are the inspiration for this project

Initial Goals

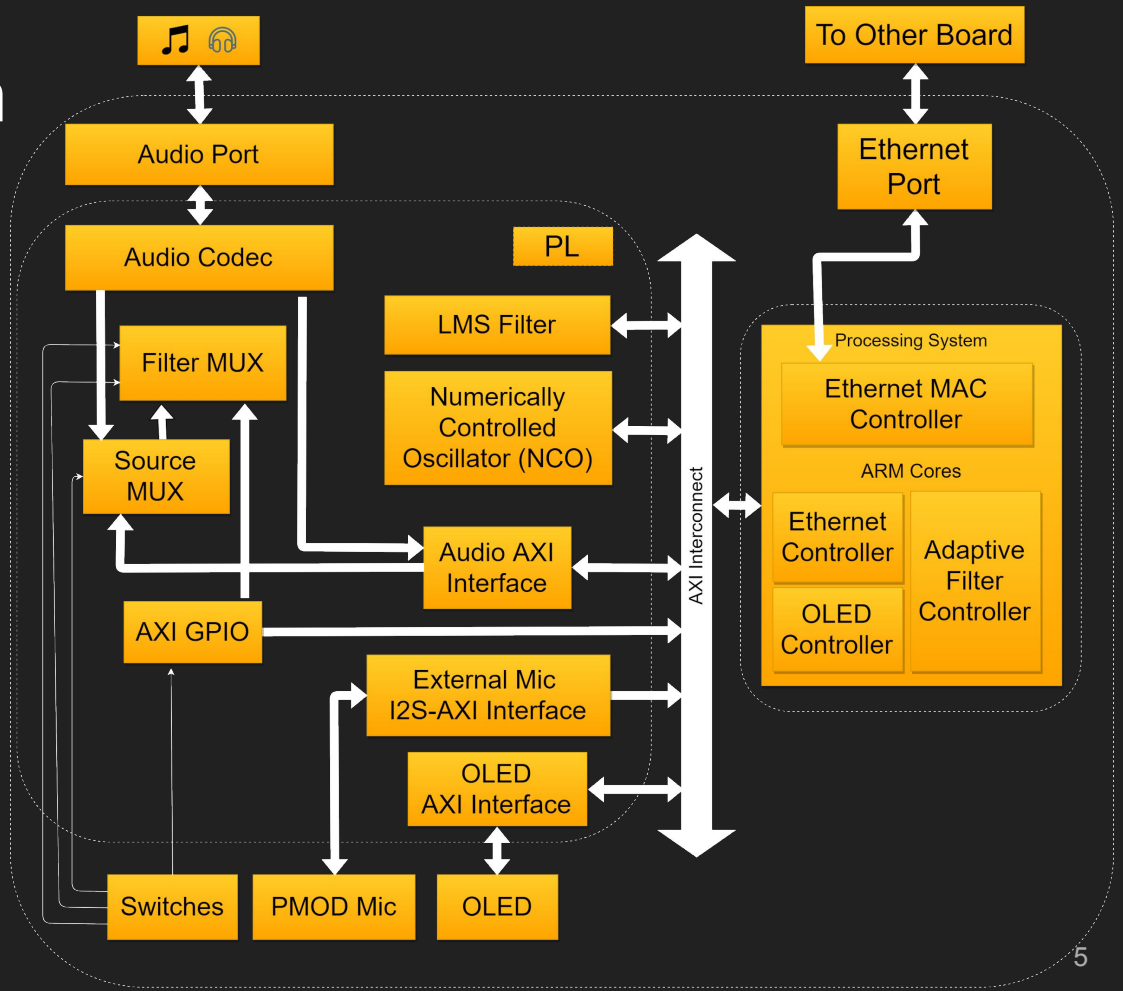
- Use LMS filter to create adaptive noise cancellation headphones.
- Two microphones
  - Primary microphone inside the headphones recording source signal + ambient noise
  - Ambient microphone outside the headphones recording ambient noise
- Adaptive filter calculates “anti-noise” waveform, which is added to signal and sent to headphones cancelling out noise
- Source signal is either Line IN or Streamed IN via Ethernet from other board’s Line IN

Final Outcome

- Successful in implementing LMS filter, however, not able to create noise cancelling headphones
- Noise leaks in from the side, PMOD mics not good enough to record it
- Two of three microphones shorted in development
  - So we changed our strategy to simulate filtering done in a pilot's headset
- Noise added to source signal
  - Via a Numerically Controlled Oscillator (NCO) delivering sine wave, or
  - Via the PMOD microphone recording ambient noise

Final System Diagram

- LMS Filter and NCO run on PL, receive inputs from PS and send outputs back to PS
- 4 AXI GPIOs
  - Move audio source data, external PMOD mic data to PS
  - Move filtered and unfiltered output to PL



LMS Adaptive Filter

Weight Update Equation

$$\hat{w}_k(n+1) = w_k(n) + \eta e(n) x_k(n)$$

Ambient Noise

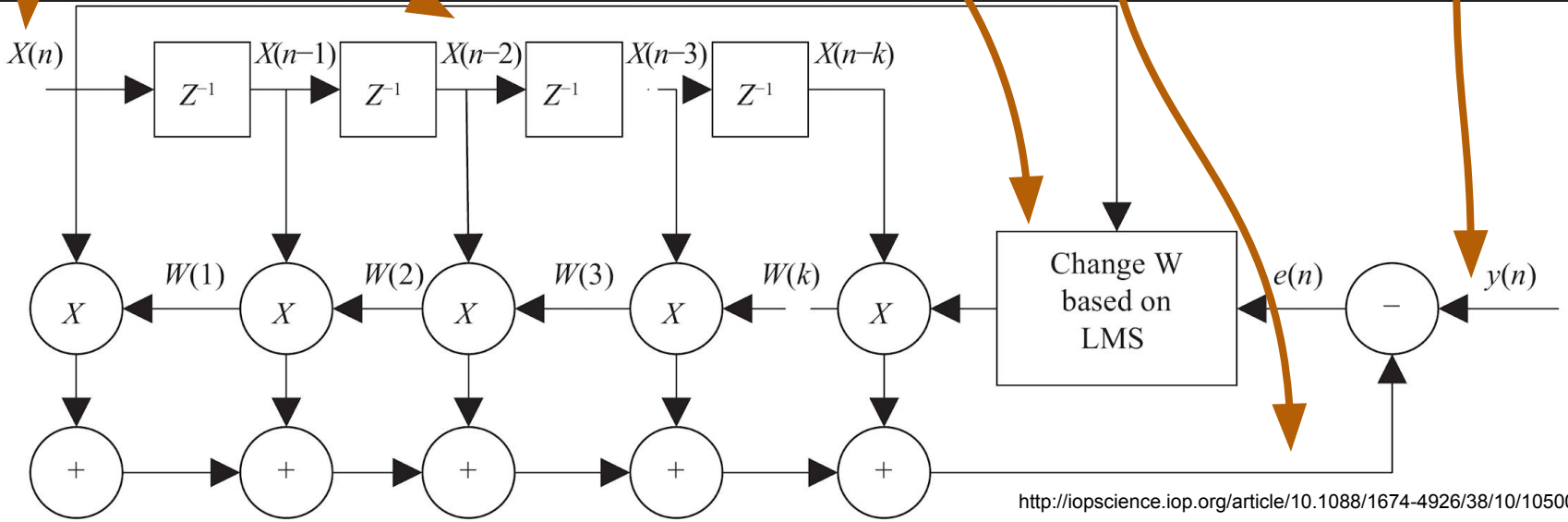
Adaptive Block implementing LMS Algorithm to update weights

Estimated Noise

Step Size

FIR Filters

Noisy Audio Source



http://iopscience.iop.org/article/10.1088/1674-4926/38/10/105003/meta

Blocks/software created

- Hardware Blocks
  - I2S interface for PMOD microphone
  - Numerically Controlled Oscillator (Via Vivado HLS)
  - LMS Filter (Via Matlab HDLCoder)

- Software Blocks
  - Ethernet UDP streaming
  - LMS, NCO AXI interface
    - Controls sending data to/from LMS filter, NCO and source multiplexer via AXI

Borrowed IP and Inspirations

- Zedboard Audio Codec
  - https://github.com/ems-kl/zedboard_audio
- Audio Code AXI Interface
  - https://github.com/Laxer3a/ZedBoardAudio
- OLED Controller
  - https://github.com/ama142/ZedboardOLED-v1.0-IP
- LMS Filter and NCO
  - Inspired by Zynq Book Tutorials (http://www.zynqbook.com/download-tuts.html)

Design process

- Tried to create a parallel working environment
  - Not working on tasks that needs edits to the same hardware or software blocks
- Simulate then synthesize (where applicable)
- Reuse existing open source IPs or tutorials
  - Helped save time debugging general issues
  - Easier to make changes to / tweak underlying VHDL code

Lessons Learned

- Hardware
  - Creating and packaging IP blocks
  - Developing an I2S interface from scratch
  - Using Simulation and Integrated Logic Analyzer to debug signals
  - Adding timing constraints, connecting external ports to FPGA pins
- Software
  - Running different C++ processes on both ARM cores at the same time
  - Sending UDP packets via Ethernet
- Hardware-Software Codesign
  - AXI Interface, connecting hardware blocks to SDK
  - Accessing and modifying GPIOs (Switches, Input/Outputs) in SDK
  - DMA to make streaming data available to Zynq for further use

Demo

- SW 0: Input Source Multiplexer
  - **0:** Line IN **1:** Stream IN
- SW 1: Enable/ Disable Filter Output
  - **0:** Source Multiplexer **1:** Filter Multiplexer (Output dependent on SW 2)
- SW 2: Filter Multiplexer
  - **0:** Unfiltered output **1:** Filtered output
- SW 7: Noise source
  - **0:** Sine wave from NCO **1:** PMOD Microphone