fLaCPGA - FPGA fLaC Decoder

Requirements Analysis

Emmanuel Jacyna - 24227498

March 21, 2016

Contents

Abstract

1 Introduction

The goal of my final year project is to produce an implementation of a Free Lossless Audio Codec (henceforth, fLaC) decoder and encoder in Verilog for eventual use on an FPGA. fLaC is a lossless audio compression codec that is gaining popularity as a method of distributing high quality audio recordings and for compressing large audio archives[1]. Whilst there are a number of software implementations of the fLaC encoder and decoder, there are no freely available hardware implementations. A hardware implementation would be of great use for a number of reasons. Many nations are now in the process of digitising large national audio archives. Audio archives require data to be compressed losslessly in order to preserve content in a form faithful to the original, however, uncompressed lossless data consumes large amounts of space. Currently,

uncompressed formats such as WAVE and (BBWAVE?) are quite popular for audio archiving[1]. Compressed audio has the major benefit of reducing space usage by up to half, doubling an archive's potential storage space. In order to convert large (terabytes) amounts of audio data to a compressed format, a lot of computing power is required. If a hardware decoder were available, it would both reduce encoding time and reduce power consumed by the encoding process. fLaC is also gaining popularity as a medium for portable audio players. The players are very sensitive to power consumption, a hardware implementation would be of great use in reducing the power load of the decoding process. Another potential use case of a hardware encoder would be in a recording studio. Instead of recording audio to an uncompressed format, high quality audio could be encoded in real time as it comes in.

2 Audio Compression Overview

Just as with general data compression, there are two main methodologies for compressing audio data, lossy and lossless. Lossy audio encoding most often includes techniques such as psychoacoustic compression, where sounds that humans cannot perceive or differentiate are removed from the audio. These techniques will not be the focus of this project. Lossless compression, by comparison, encodes audio perfectly, that is, the decoded audio is identical to the original input samples. Audio is still affected by problems such as quantisation noise and recording device noise.
The main goal of any lossless compression algorithm is to remove redundancy from data, thereby reducing the amount of information needed to reproduce the data. Audio data is often highly redundant, samples of data that are close to each other will usually have very similar patterns, for example, samples of a clarinet playing the same note for a number of seconds will clearly have a similar spectral pattern over the period of time the note is held, thus offering redundancy to be exploited. The most popular method for lossless compression of audio data is to find an accurate model of the audio (a *predictor*), find the error (often called *residuals*) between the predictor and the true audio, then to encode the residuals using a variable bit length encoding in order to reduce the redundancy of the signal. The fLaC lossless audio compression algorithm borrows heavily from prior work in lossless encoding including the Shorten algorithm[**?**], and the AudioPAK algorithm[2]. These algorithms use a technique called *linear prediction* to produce their audio model, and use Rice encoding to perform the entropy encoding phase of the compression. The advantage of these algorithms is the ease with which they can be translated into hardware, as the linear prediction step consists of a number of multiplies and adds.

3 Requirements

3.1 Documentation

A number of documents will be produced during this p

- Create a literature review
- Create final report
- Create a project description poster
- Encoder and Decoder documentation, including in code documentation (comments) and a user guide

3.2 fLaC Decoder

- Implement software fLaC decoder in C++
- Implement fLaC decoder in Verilog
- Develop Verilog testbenches for each module
- Test decoder on FPGA
- Produce audio output using Development Board DAC

3.2.1 Metadata and Frame Decoding

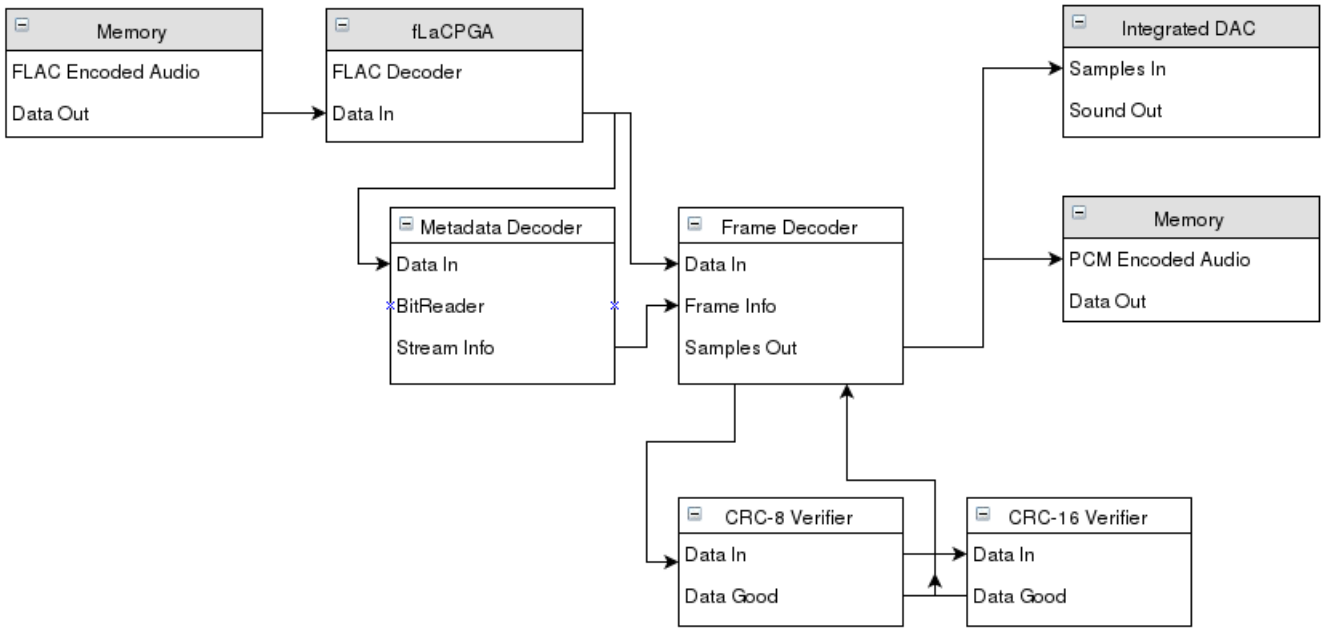

Figure 1: Overview of fLaCPGA Decoding System

- Decode fLaC Metadata stream
  - Read and store variable bit length fields

  - Read STREAMINFO metadata block
  - Read and ignore all other metadata blocks

- Decode fLaC Frame
  - Read and store variable bit length fields
  - Execute a CRC-8 check on the frame header
  - Execute a CRC-16 check on the entire frame
  - Decode each subframe

3.2.2 Subframe Decoding

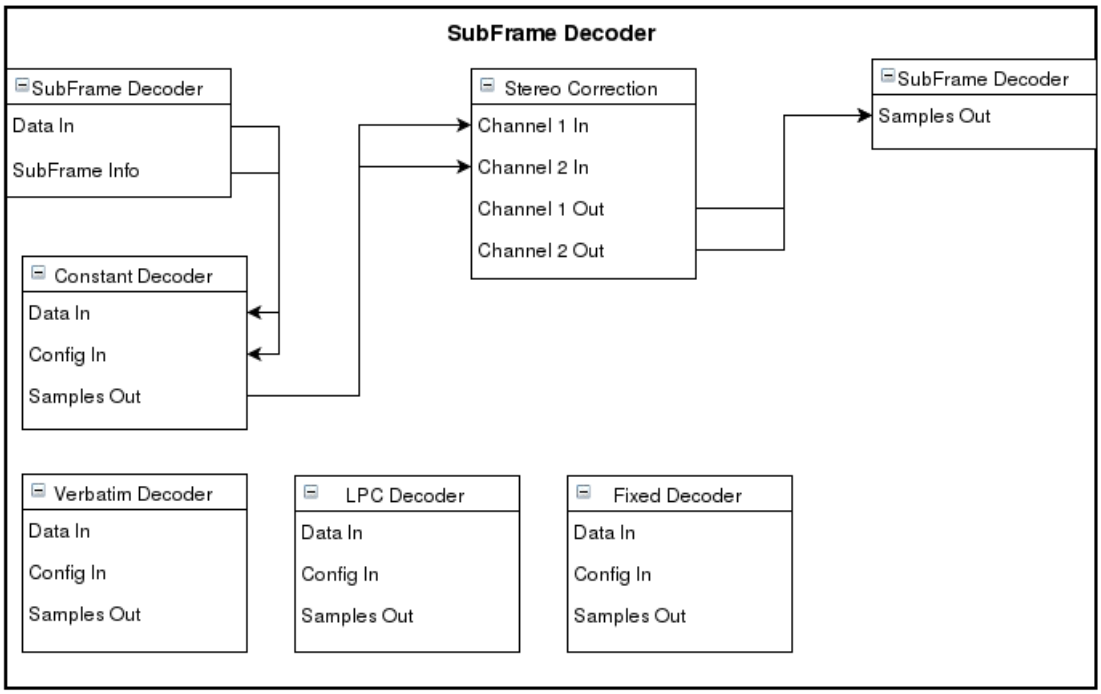

Figure 2: Overview of fLaCPGA Subframe Decoding System

- Decode fLaC Subframe
  - Decode Constant encoded subframe
  - Decode Verbatim encoded subframe
  - Decode Fixed encoded subframe
  - Decode LPC Encoded subframe
  - Correct stereo decorrelation

3.2.3 LPC and Fixed Decoding

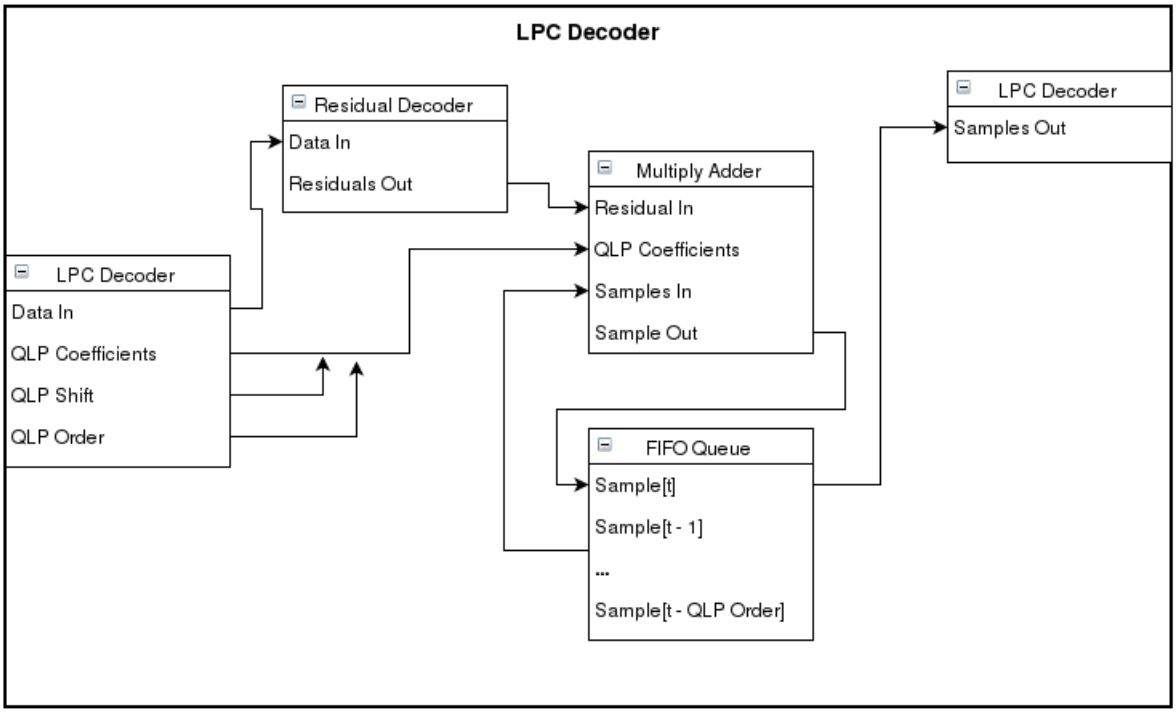

Figure 3: Overview of fLaCPGA LPC Subframe Decoding System

- Decode LPC and Fixed Subframes
  - Read and store variable bit length fields
  - Decode fLaC Specification residual
    - Decode rice encoded variable length bit stream
  - Perform LPC decoding using residuals and predictor coefficients

3.3 fLaC Encoder

- Implement software fLaC encoder in C++
- Implement fLaC encoder in Verilog
- Develop Verilog testbenches for each module
- Test encoder on FPGA

4 Conclusion

References

[1] Find Citation. Find citation. *Find Citation*.

[2] M. Hans and R. W. Schafer. Lossless compression of digital audio. *IEEE Signal Processing Magazine*, 18(4):21–32, Jul 2001.