AudioSynthWavetable Design Specification

**Project: TeensyAudio Wavetable Synthesis**

Authors: Ryan Mellmer

Connor Delaplane

Aida Keifer

Nicholas Craig

Josh Bucklin

Jonathan Jensen

Xuan Tang

Date: 2016-11-28

Version: 1.0

**Table of Contents**

[**1** **INTRODUCTION**](#_3znysh7)

[1.1 Objectives](#_2et92p0)

[1.2 Scope](#_tyjcwt)

[**2** **Functionality**](#_3dy6vkm)

[2.1 Site Map](#_4d34og8)

[**3** **Architecture**](#_17dp8vu)

[3.1 Architecture Overview](#_3rdcrjn)

[3.2 Application Interfaces](#_35nkun2)

[**4** **Design**](#_1ksv4uv)

[4.1 Object Diagram](#_44sinio)

[**5** **CODE**](#_4i7ojhp)

[5.1 File Names and Structure](#_2xcytpi)

[5.2 Executable File Names and Structure](#_1ci93xb)

# INTRODUCTION

## Objectives

The purpose of this document is to outline the technical specifications for the Wavetable Synthesis library to be used on the Teensy 3.2 microcontroller. It is primarily meant as a guide for the developers of the library and will cover the following areas:

* Definitions of all objects within the library along with their interfaces.
* Interactions with the existing Audiostream.h library API.

## Scope

The purpose of this project is to provide a C++ library and accompanying Python utility scripts allowing realistic instrumentation audio to be synthesized on the Teensy 3.2 Arduino Digital Analog Converter (DAC). This library will be exposed to developers, and will allow pitch shifting, looping, tremolo, and vibrato effects to be imposed on a raw byte buffer of recorded samples.

# Functionality

## Soundfont Decoding

The basic functionality of the SoundFont Decoding script is defined in the Wavetable Synthesis on Microcontroller Requirements document. A brief summary is provided here.

* User executes the script in a console
* User loads a .sf2 file into the utility by passing it’s location to the script
* If the .sf2 adheres to pre-defined limitations for what the Wavetable Library accepts, then allow the user to select which instrument to import
* Parse samples into logical units
* Output the resulting samples into a C++ struct/object within a .cpp file for use within the Wavetable Synthesis Library

## Wavetable Synthesis Library

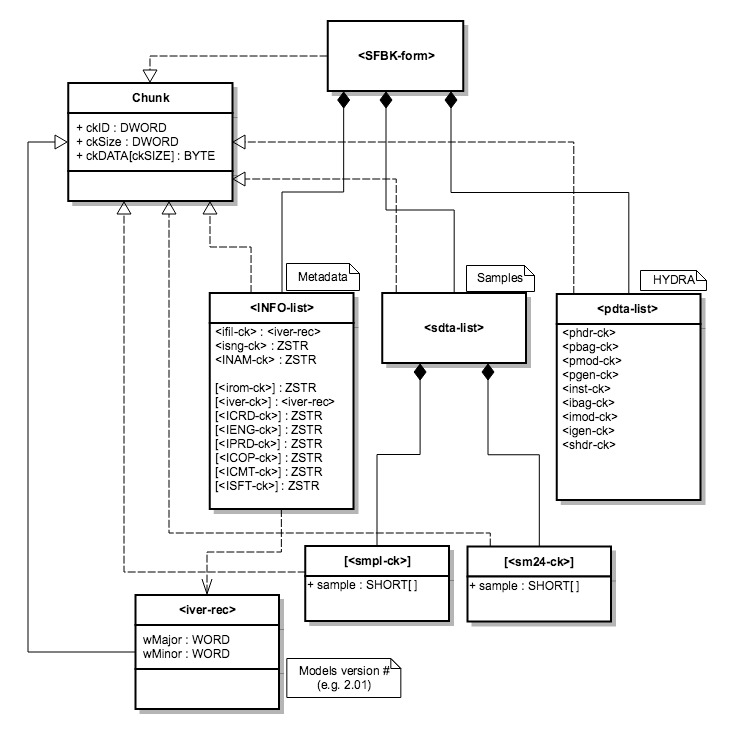
The basic functionality of the Wavetable Synthesis Library is defined in the Wavetable Synthesis on Microcontroller Requirements document. A brief summary is provided here.

* Load audio data from the file produced by SF2 extraction process
* Play silence in the event of read error
* Create interpolated audio data on the Teensy
* An audio sample’s sustain data can be looped
* Audio data can be modified with a tremolo effect
* Audio data can be modified with a vibrato effect

# Architecture

## Architecture Overview

**SF2 file object diagram:**

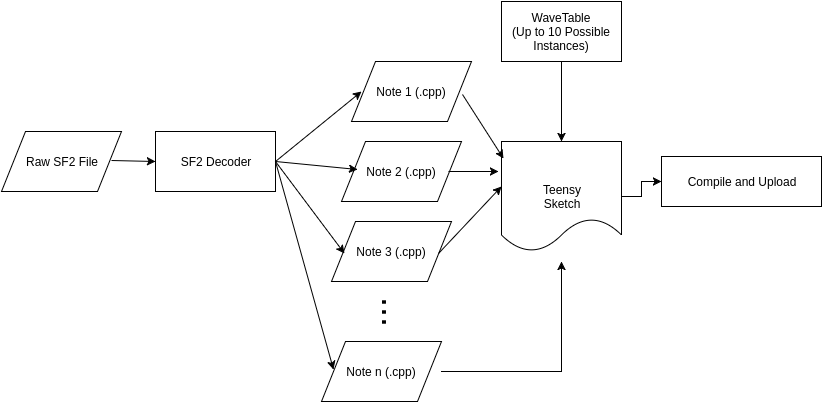


**SoundFont Decoding:**



**Teensy C++ Library:**

This diagram gives a general overview of how both the SF2 Decoder and the Teensy C++ Library will work together. It’s important to notice that a single Note # file generated by the SF2 Decoder will contain all three phases for a single sample (attack, sustain, release). In addition, while there is only a single arrow from the Wavetable to Teensy Sketch there could be up to 10 instances. There should be at least one Wavetable object for every note included in the Teensy Sketch file.



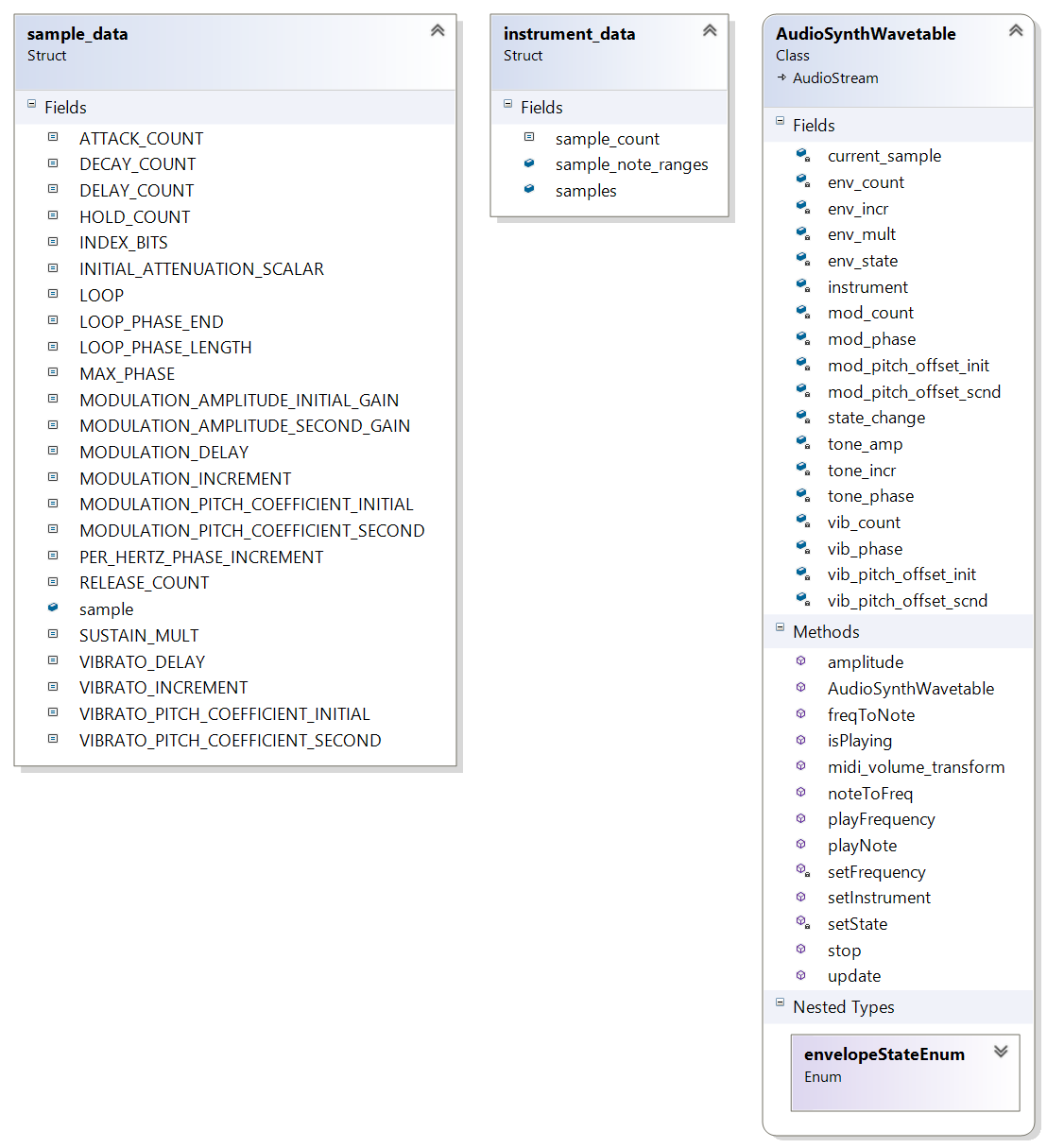
## Application Interfaces

This section contains a description of all SF2 Decoding and Teensy C++ Library application interfaces.

**Python SF2 Decoding:**

The SoundFont decoding script will be ran only through the command-line, so the only interface to be described is the command-line. The user will need to input the location of the SoundFont file to decode, and the script will walk the user through the decoding process.

**Teensy C++ Library:**



The AudioSynthWavetable library is made up of 3 distinct datatypes:

1. AudioSynthWavetable Class object
2. instrument\_data struct
3. sample\_data struct

The AudioSynthWavetable object is the main API object to interface with in order to utelize the wavetable library. This class exposes methods surrounding playing notes, changing frequency of playback, stopping notes, and setting the instrument to play. A complete description of the AudioSynthWavetable class can be found at <https://teensyaudio.github.io/Wavetable-Synthesis/html/index.html>

The instrument\_data struct contains information which makes up a complete instrument, such as an array of samples in the form of sample\_data structs, the number of samples for the instrument, and key range upper bounds for each sample within the instrument. The sample\_data struct contains information surrounding a specific sample within an instrument. The sample\_data information is exported and stored as algebraic expressions as to optimize and reduce the amount of processing that has to be done on the Teensy at runtime as to increase playback efficiency. A full listing of all of the fields within these structs as well as the purpose for their values can be found within the sample\_data.cpp file.

# Design

## Class Diagrams

**Soundfont Decoding:**

The SoundFont decoding script will be contained within a single Python file. Since the script will be a purely command-line driven tool, an open source library, sf2utils, will be utilized to assist in the soundfont decoding process.

**Teensy C++ Library:**

The Wavetable Synthesis library consists of one primary class called AudioSynthWavetable. A class diagram that lists the fields and methods of this class is given above, and documentation for the API specifics is given at <https://teensyaudio.github.io/Wavetable-Synthesis/html/index.html>

This is followed by a brief description of each element within the diagram. Note: The public methods and their functionality are listed in section 3.2, Application Interfaces.

# CODE

## File Names and Structure

This section lists the name, purpose and structure of all expected files for this project.

**SoundFont Decoding Utility:**

**File #1:** decoder.py

**Purpose:** Main Python script file which the user will actually interact with to decode samples from a soundfont file.

**Structure:** This file will contain the entirety of the necessary functionality for the SF2 decoding script.

**Teensy C++ Library:**

**File #1:** AudioSynthWavetable.h

**Purpose:** This file contains the Wavetable class prototype and is intended to be #include(d) within any file using the Wavetable object.

**Structure:** This file will contain a single class prototype, Wavetable, that is derived from the AudioStream class. As such the prototype will contain the update() method along with all other fields and methods listed in the Wavetable class diagram found in section 4.1.

In addition to the class prototype the Audiostream.h library will be #include(d) so it is available for the new Wavetable class.

**File #2:** AudioSynthWavetable.cpp

**Purpose:** This file contains the implementation of all of the Wavetable class methods.

**Structure:** This file will #include the Wavetable.h file and will contain implementations for all public and private methods of the class.