Министерство науки и высшего образования Российской Федерации

Новгородский государственный университет имени Ярослава Мудрого

Кафедра ИТИС

Функциональная спецификация

VSTi Синтезатор

Версия 0.0.1

Выполнил:

ст. гр. 8091

Васильев Иван  
Владимирович

Проверил:

Преподаватель Макаров В.А.

Дата

Великий Новгород

2021

# **Введение**

Функциональная спецификация разработана по проекту “VSTi Синтезатор”. Данный документ содержит необходимый набор согласованных требований, на основании которых будет разработан программный продукт “VSTi Синтезатор”.

# **Назначение и область применения**

Продукт предназначен для генерации звуковых сигналов в области программных средств звукового редактора. Аудитория пользователей состоит из заинтересованных лиц в написание музыкальных партий и исследованию принципа работы синтеза звука.

# **Цель разработки**

Получение практических и теоретических навыков в области разработки музыкального программного продукта, результатом которого является встраиваемый модуль для цифровых программных средств обработки звука.

# **Термины и сокращения**

|  |  |  |
| --- | --- | --- |
| Сокращение | Термин | Определение |
| DAW | digital audio workstation | компьютерная система, предназначенная для [записи](https://ru.wikipedia.org/wiki/%D0%97%D0%B2%D1%83%D0%BA%D0%BE%D0%B7%D0%B0%D0%BF%D0%B8%D1%81%D1%8C), хранения, редактирования и воспроизведения [цифрового звука](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B7%D0%B2%D1%83%D0%BA%D0%BE%D0%B7%D0%B0%D0%BF%D0%B8%D1%81%D1%8C). |
| VSTi | Virtual Studio Technology instrument | программный интерфейс аудио-плагина, который интегрирует [программные синтезаторы](https://en.wikipedia.org/wiki/Software_synthesizer) и [блоки эффектов](https://en.wikipedia.org/wiki/Effects_unit) в [цифровые аудио-рабочие станции](https://en.wikipedia.org/wiki/Digital_audio_workstation).  (стандарт [VST2](https://steinberg.help)) |
| MIDI | Musical Instrument Digital Interface | [стандарт](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82) [цифровой звукозаписи](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%BE%D0%B9_%D0%B7%D0%B2%D1%83%D0%BA) на формат обмена данными между [электронными музыкальными инструментами](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5_%D0%BC%D1%83%D0%B7%D1%8B%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B8%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D1%8B). (Стандарт [General MIDI MMA](http://www.midi.com)) |
| [ADSR](https://ru.wikipedia.org/wiki/ADSR) | Attack-Decay-Sustain-Release | функция, описывающая изменения параметров (Attack, Decay, Sustain, Release) во времени. |
|  | Дисторшн | [звуковой эффект](https://ru.wikipedia.org/wiki/%D0%97%D0%B2%D1%83%D0%BA%D0%BE%D0%B2%D0%BE%D0%B9_%D1%8D%D1%84%D1%84%D0%B5%D0%BA%D1%82), достигаемый искажением сигнала путём его «жёсткого» ограничения по [амплитуде](https://ru.wikipedia.org/wiki/%D0%90%D0%BC%D0%BF%D0%BB%D0%B8%D1%82%D1%83%D0%B4%D0%B0). |
|  | Дилей | [звуковой эффект](https://ru.wikipedia.org/wiki/%D0%97%D0%B2%D1%83%D0%BA%D0%BE%D0%B2%D0%BE%D0%B9_%D1%8D%D1%84%D1%84%D0%B5%D0%BA%D1%82), имитирующее чёткие затухающие повторы ([эхо](https://ru.wikipedia.org/wiki/%D0%AD%D1%85%D0%BE)) исходного сигнала |

# **Описание структурной модели программы**

* 1. **Структурные модели: (модули и их взаимодействия)**
     1. Структурная модель объектов программы:
* Класс AudioProcessor является главным звеном в генерации и обработки звукового сигнала и имеет следующие методы:
  + generateSignal() – запускает генерацию сигнала
  + processEffects() – запускает цепочку обработки сигнала следующими эффектами: частотный фильтр, дисторшн и дилей согласно пунктам 5.1.3.4, 5.1.3.5, 5.1.3.6 из документа ТЗ.
  + outSignal() – функция возвращающая пользователю сгенерированный и обработанный звуковой сигнал
* Класс Generator осуществляет генерацию звуковых сигналов в параллельных каналах и функцию смешивания этих двух каналов в один, согласно пунктам 5.1.3.1 и 5.1.3.3 из документа ТЗ. Класс имеет следующие методы:
  + generateFirstSignal() – генерирует первый звуковой сигнал
  + generateSecondSignal() – генерирует второй звуковой сигнал
  + mergeSignal() – смешивания двух сигналов в один
* Класс Oscillator реализует генерацию звукового сигнала согласно пунктам 5.1.3.1.1 - 5.1.3.1.4 из документа ТЗ. Класс имеет следующие методы:
  + createWaveSinus() – осуществляет генерацию сигнала типа “синус”
  + createWaveTriangle() – осуществляет генерацию сигнала типа “треугольник”
  + createWaveSquare() – осуществляет генерацию сигнала типа “треугольник”
  + createWaveSaw() – осуществляет генерацию типа “Пила”
  + makeEnvelope() – осуществляет обработку сигнала ADSR огибающей
* Класс ADSR реализует обработку звукового сигнала с помощью ADSR огибающей согласно пункту 5.1.3.2 из документа ТЗ. Класс имеет следующий метод:
  + ADSR() – реализация обработки ADSR огибающей
* Класс Filter реализует обработку звукового сигнала при помощи звукового частотного фильтра согласно пункту 5.1.3.4 из документа ТЗ. Класс имеет следующие методы:
  + convolution() – свёртка
  + highPass() – подсчёт коэффициентов для ВЧ фильтра
  + lowPass() – подсчёт коэффициентов для НЧ фильтра
* Класс Distortion реализует обработку звукового сигнала при помощи эффекта дисторшн согласно пункту 5.1.3.5 из документа ТЗ. Класс имеет следующий метод:
  + distortionProcessing() – реализации эффекта дисторшн
* Класс Delay реализует обработку звукового сигнала при помощи эффекта дилей согласно пункту 5.1.3.6 из документа ТЗ. Класс имеет следующий метод:
  + delayProcessing() - реализации эффекта дилей
* Класс Window осуществляет работу с пользователем, что включает в себя пользовательское окно согласно пункту 5.2.1 из документа ТЗ. Класс имеет следующие методы:
  + showWindow() – показывает пользовательское окно
  + saveConfgInParams() – сохраняет введённые значения из пользовательского окна
* Класс Master реализует мастер канал и регулирует громкость выходного сигнала согласно пункту 5.1.3.7.1 из документа ТЗ. Класс имеет следующие методы:
  + adjustingVolume() – регулировка выходной громкости сигнала
  + outSignalUser() – вывод сигнала в DAW

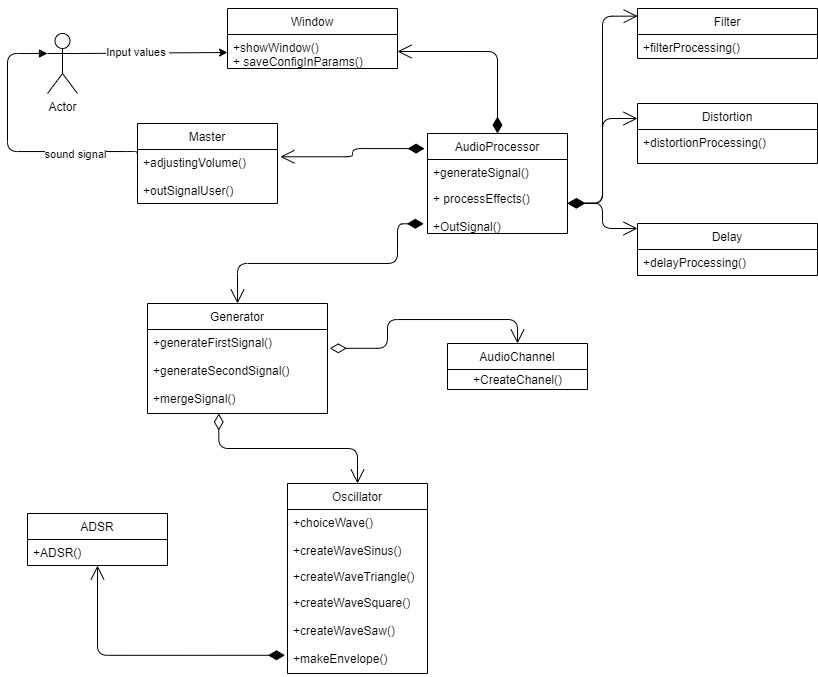


Рисунок 1 Структурная модель объектов программы

* 1. **Алгоритмические модели: (описывает интересные алгоритмы программы)**
     1. Алгоритм частотного фильтра
* За основу был взят фильтр Баттерворта, благодаря его максимально гладкой АЧХ на полосах пропускания.

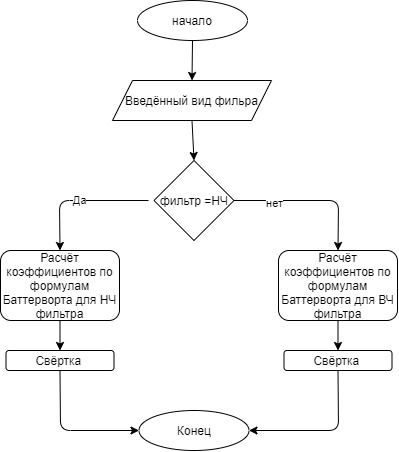


Рисунок 2 Блок-схема алгоритма работы частотного фильтра

* Алгоритм работы фильтра сводится к свертке, которая делается последовательно для каждого семпла:

y(n) — это новое значение семпла, которое нужно рассчитать. x(n) — текущее значение семпла, соответственно y(n-1) и y(n-2) — предыдущие 2 рассчитанных семпла, а

x(n-1) и x(n-2) — предыдущие входные значения семплов.

* + 1. Алгоритм обработки дисторшн
* Дисторшн определяется только параметром максимального абсолютного значения амплитуды — Threshold. Абсолютные значения семпла не превосходят 1, значит и Threshold заключен в интервал [0,1].

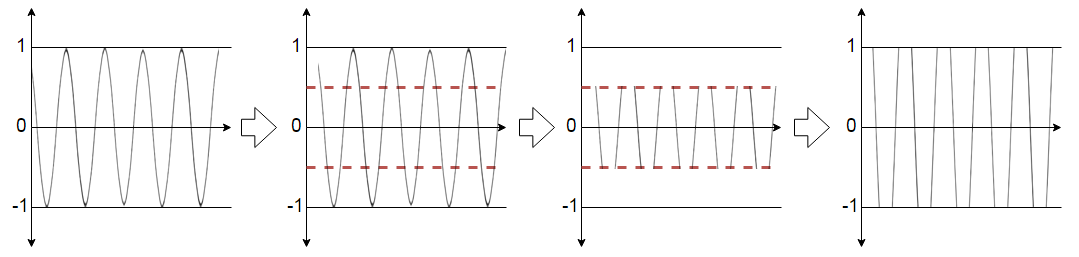


Рисунок 3 Алгоритм работы дисторшн на сигнале

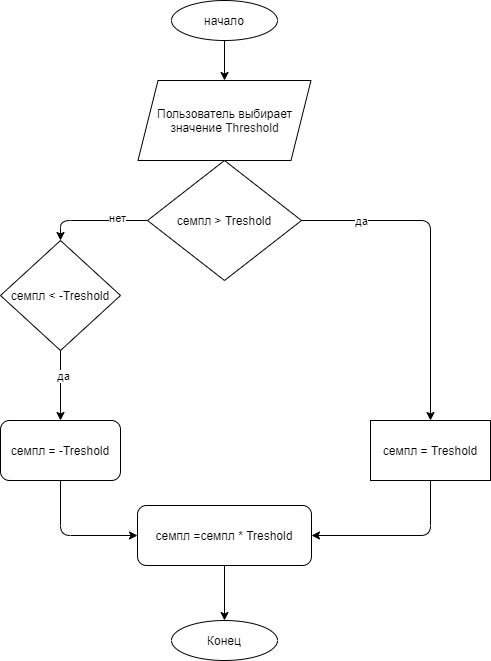


Рисунок 4Блок-схема алгоритма работы дисторшн

Заказчик: Исполнитель:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

м.п. м.п.