Научно-исследовательский университет

Московский энергетический институт

Кафедра

Вычислительных машин, систем и сетей

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

Отчет

по производственной практике

на предприятии ФГБОУ ВО НИУ «МЭИ»

Студент: Коваль Роман\_\_\_\_\_\_\_\_\_

Группа: А-08-17 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики:

Подпись:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: «8» июля 2020 г.\_\_\_\_\_\_\_\_\_

Москва 2020 г.

**Введение**

**Постановка задачи**

Задача: написать программу, работающую с синтезатором в составе звуковой карты по протоколу MIDI при помощи функций Win32 API (стандартных функций Windows).

Требуется разработать приложение, которое бы использовало только встроенный средства Windows API для работы с MIDI-устройством, встроенным по умолчанию в звуковую карту компьютера. Для этого нужно изучить теоретические материалы, такие как:

* Протокол MIDI – соглашения, схемы, по которым происходит передача информации от устройства к устройству. Спецификации также описывают формат сообщений, аппаратный и прикладной уровни.
* Интерфейс взаимодействия прикладной программы с MIDI-устройством через систему (Windows API). Функции, реализующие данный интерфейс.
* Формат файлов .MID, которые содержат определенные команды, позволяющие системе взаимодействовать с MIDI-устройством.

Изученный теоретический материал позволит разработать приложение, с помощью которого пользователь сможет общаться с синтезатором в составе звуковой карты, сможет проигрывать .MID файлы, а также создавать такие файлы.

**Объект задания**

Непосредственным объектом задания является приложение, которое использует интерфейс Windows API для взаимодействия с MIDI-устройством (синтезатором) в составе звуковой карты.

Исходя из поставленной задачи, можно изобразить схему объекта задания, описывающую интерфейсы и каналы взаимодействия потребителя приложения (пользователя) с конечным MIDI-устройством. Схема, которую должно реализовывать приложение, представлена на рис. 1.



*Рис. 1. Схема объекта задания*

**Метод решения задачи**

Решение задачи заключается в написании кода на языке программирования, который позволил бы реализовать схему объекта задания.

Исходя из поставленной задачи и описания объекта задания, формируются требования к методу решения задачи:

* Среда исполнения приложения: 32/64-разрядная Windows (интерфейс Win32 API поддерживается с версии Windows NT)
* IDE для построения проекта приложения и его разработки (включая использование ресурсов, компиляцию, подсветку синтаксиса, инструменты рефакторинга): Microsoft Visual Studio
* Наличие MIDI-устройства в составе звуковой карты: на данный момент в большинстве случаев уже встроен синтезатор Microsoft GS Wavetable Synth
* Язык программирования, удовлетворяющий выше описанным требованиям: C/C++, Delphi

На основании требований к методу решения задачи и уровня навыков был выбран язык программирования C++ (стандарта C++17), который поддерживается IDE Microsoft Visual Studio и позволяет создавать приложения для Windows с использованием Win32 API.

Для отладки и тестирования приложения в наличии имеются устройства, работающие на ОС Windows 7 и Windows 10 и имеющие в составе своей звуковой карты синтезатор Microsoft GS Wavetable Synth.

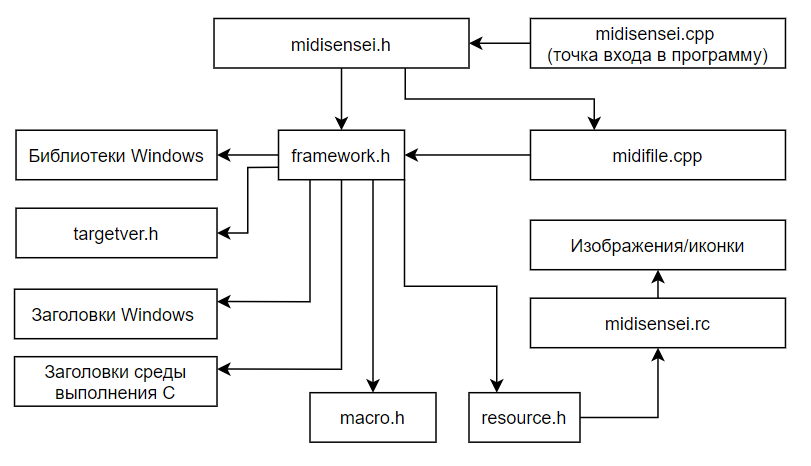
Таким образом, методом решения задачи является разработка приложения в IDE Microsoft Visual Studio на языке программирования C++ для выполнения в средах Windows 7/Windows 10 и наличием синтезатора Microsoft GS Wavetable Synth в составе звуковой карты.

**Экспериментальная часть (разработка приложения)**

Для решения поставленной задачи, после изучения теоретических материалов, требуется применить описанный метод решения.

Первый этап разработки приложения заключается в проектировании приложения и создании каркаса проекта в IDE Microsoft Visual Studio.

Структура проекта приложения представлена на рис. 2.



*Рис. 2. Структура проекта приложения*

Подробнее о каждой единице структуры проекта:

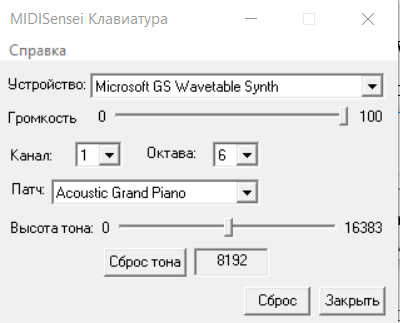
1. **midisensei.cpp** – основный исходный файл приложения, точка входа в программу
2. **midisensei.h** – файл заголовка для основного исходного файла, служит интерфейсом включения файла фреймворка и расширения для работы с MIDI-файлами
3. **midifile.cpp** – расширение для работы с MIDI-файлами (инкапсуляция структур и методов работы с файлами от основой логики приложения)
4. **framework.h** – файл заголовка для включения общих для всего проекта файлов, таких как заголовки среды выполнения C и заголовки Windows
5. **библиотеки Windows** – библиотечный файлы типа .lib и .dll, предоставляемых системой Windows (например, мультимедиа winmm.lib)
6. **targetver.h** – заголовок используемых средств разработчика (SDK) для целевой платформы
7. **заголовки Windows** – заголовочные файлы, предоставляющие интерфейс работы с ОС (WinAPI; например, mmeapi.h – работа с мультимедиа)
8. **заголовки среды выполнения C** – заголовочные файлы, предоставляемые решения стандартной библиотеки (stdlib.h, thread, …)
9. **macro.h** – заголовок, содержащий макросы, используемые в приложении
10. **resource.h** – заголовок, содержащий макросы для работы с ресурсами
11. **midisensei.rc** – описание используемых ресурсов в приложении (диалоговые окна, кнопки, меню)
12. **изображения/иконки**

Следующий этап разработки – реализация первой функции приложения – работа с синтезатором, MIDI-клавиатура.

Было разработано немодальное[[1]](#footnote-1) диалоговое окно, предоставляющее следующую функциональность:

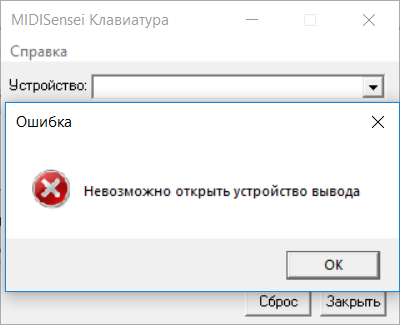
* выбор MIDI-устройств из списка (включая MIDI Mapper)
* трекбар громкости MIDI-устройства (0-100)
* выбор канала (1-16)
* выбор октавы (0-10)
* выбор патча/тембра/инструмента (все доступные 128 значений)
* трекбар «Высота тона» - так называемый Pitch Wheel (0-16383, с отображением текущего значения и возможностью сброса до значения по умолчанию)
* отслеживание нажатия клавиш клавиатуры с воспроизведением соответствующей ноты (с учетом канала, октавы, высоты тона и текущего патча)
* сброс устройства (при «залипании» нот)

Графический интерфейс разработанного диалогового окна представлен на рис. 3.



*Рис. 3. Диалоговое окно «MIDISensei Клавиатура»*

Также была предусмотрена обработка ошибок во избежание неопределенного поведения и выброса исключений. Пример обработанной ошибки представлен на рис. 4.



*Рис. 4. Обработка ошибок в диалоговом окне «MIDISensei Клавиатура»*

Третьим этапом разработки является реализация второй функции приложения – открытие и проигрывание .MID файлов.

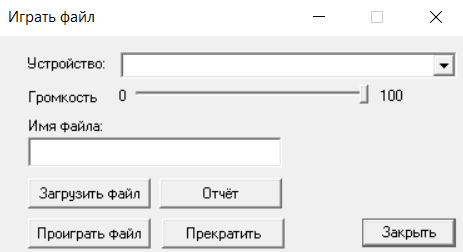
В ходе данного этапа с помощью оконного интерфейса Win32 API и событийно-ориентированной парадигме программирования было разработано модальное диалоговое окно, которое позволяет пользователю:

* выбрать устройство вывода из списка (аналогично окну Клавиатуры)
* установить громкость устройства вывода (0-100)
* загрузить .MID файл
* просмотреть отчёт (информацию) о последнем загруженном .MID файле (пример отчёта в Приложениях)
* начать воспроизведение файла (поддерживаемые форматы: 0 и 1; стоит заметить, что фактически формат 2 почти нигде не используется); воспроизведение мульти-трекового .MID файла реализовано с использованием параллельного программирования средствами стандартной библиотеки C++
* немедленно прекратить проигрывание файла

Также была реализована обработка следующих ошибок при работе с данным диалоговым окном:

* не удалось открыть устройство вывода
* не удалось загрузить файл
* отсутствует файл отчёта
* попытка загрузить новый файл в момент воспроизведения
* попытка проиграть файл в момент воспроизведения

Графический интерфейс диалогового окна представлен на рис. 5.



*Рис. 5. Диалоговое окно «Играть файл»*

Следующий этап разработки представляет из себя реализацию третьей (последней) функции целевого приложения – создание .MID файлов.

В результате данного этапа было разработано модальное диалоговое окно, содержащее следующий функционал:

* выбор устройства вывода (аналогично предыдущим диалоговым окнам)
* трекбар громкости устройства вывода (0-100)
* обязательное поле BPM[[2]](#footnote-2)
* поле Размер такта (в формате нумератор/деноминатор), по умолчанию 4/4
* обязательное поле Тиков на четверть[[3]](#footnote-3)
* управление треками – возможность добавить/удалить трек и проиграть/прекратить трек
* управление событиями: представлены все основные события (кроме мета-событий и SysEx); выбор типа события, канала, задержки перед событием (delta-time) в миллисекундах, первый байт данных и второй байт данных (если нужен); возможность вставить событие в трек перед выбранным или удалить выбранное
* создание .MID файла из составленных треков
* сброс всех текущих треков (кнопка «Сброс»)

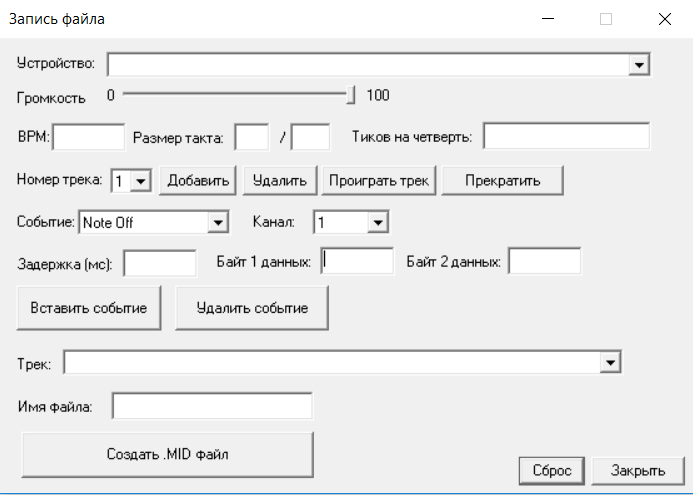
Также для данного диалогового окна была проработано обработка следующих ошибок:

* не удалость открыть устройство вывода
* не заполнено поле BPM
* не заполнено поле Тиков на четверть
* попытка проиграть трек во время воспроизведения
* не удалось создать файл
* неверный диапазон для однобайтовых значений (доступный 0..127)

Графический интерфейс разработанного диалогового окна представлен на рис. 6.

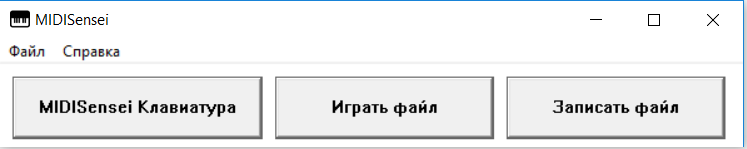
Предпоследним этапом разработки является создание главного окна приложения и разработка интерфейса к уже разработанным диалоговым окнам, предоставляющим основную функциональность объекта задания.

В ходе данного этапа было разработано главное окно приложения, которое содержит три кнопки: MIDISensei Клавиатура, Играть файл, Запись файла, предоставляющих доступ к соответствующим диалоговым окнам. При этом так как диалоговое окно «MIDISensei Клавиатура» является немодальным, была добавлена дополнительная логика, блокирующая кнопки на главном окне при активной MIDI-клавиатуре.



*Рис. 6. Диалоговое окно «Запись файла»*

Графический интерфейс главного окна приложения представлен на рис. 7. При запуске приложения окно появляется в центре экрана и запускает диалоговые окна также в центре экрана.



*Рис. 7. Главное окно приложения MIDISensei*

Последним этапом разработки приложения является отладка и тестирование разработанного функционала.

Для качественного выполнения данного этапа были подготовлены и успешно проверены следующие тестовые единицы:

1. Позиционирование окон (центральная точка)
2. Отсутствие влияния диалоговых окон друг на друга
   1. Отсутствие перекрестных свойств (за исключением громкости устройства – намеренная функциональность)
   2. Отсутствие попыток использовать одновременно одно устройство
   3. Невозможность запустить несколько диалоговых окон
   4. Успешное многократное открытие и закрытие (в разной последовательности) диалоговых окон
3. Диалоговое окно «MIDISensei Клавиатура»
4. Корректная обработка ошибок
5. Работа трекбара громкости
6. Воспроизведение всех нот (на соответствующих октаве, канале, патче)
7. Работа трекбара высоты тона
8. Кнопка «Сброс»
9. Диалоговое окно «Играть файл»
10. Корректная обработка ошибок
11. Работа трекбара громкости
12. Загрузка файлов и корректность составленного отчета
13. Проигрывание файлов формата 0 и 1
14. Прекращение проигрывания файла
15. Диалоговое окно «Записать файл»
16. Корректная обработка ошибок
17. Работа трекбара громкости
18. Управление треками
19. Управление различными событиями
20. Проигрывание трека
21. Прекращение проигрывания трека
22. Создание файла
23. Кнопка «Сброс»
24. Проигрывание созданного файла с помощью других MIDI-проигрывателей (Windows Media Player, MIDI Player BSE)
25. Проигрывание созданного файла и корректность параметров (по отчёту) с использованием функционала диалогового окна «Играть файл»
26. Корректное завершение программы
27. Корректная работа программы как в ОС Windows 7, так и в Windows 10

Полный листинг исходных файлов разработанного приложения представлен в Приложениях.

**Выводы по экспериментальной части**

Было разработано приложение – объект задания – с применением задуманного метода решения, которое удовлетворяет поставленной задаче. Приложение позволяет пользователю:

* Работать с синтезатором в составе звуковой карты через графический интерфейс: играть ноты всех октав, применять патчи, менять высоту тона, менять канал и громкость; программа, в свою очередь, передаёт команды пользователя операционной системе через функции интерфейса Win32 API, а операционная система управляет MIDI-устройством согласно заданному протоколу
* Открывать и проигрывать .MID файлы; при этом был использован подход параллельного программирования (в контексте параллельных потоков), что позволило воспроизводить успешно не только файлы формата 0 (однотрековые), но и файлы формата 1 (мульти-трековые)
* Создавать .MID файлы с гибкими параметрами (темп, количество тиков на четверть, различная задержка, множество треков, все доступные события, за исключением meta и SysEx), которые можно затем проиграть как через само приложение, так и через сторонний MIDI-проигрыватель (например, Windows Media Player).

Получившееся приложение получилось сравнительно положительным в отношении пользовательского опыта, имеет законченный функционал (который, тем не менее, не лишен перспективы доработок и расширения), а также реализует обработку ошибок, позволяющую предотвратить неопределенное поведение и неожиданные исключения.

По итогам отладки и тестирования был сделан вывод, что приложение способно корректно выполнять заложенную функциональность, работать неограниченное время без сбоев и корректно завершаться.

**Список использованных источников**

* 1. The audio programming book / edited by Richard Boulanger and Victor Lazzarini; foreword by Max Mathews. – Cambridge, Massachusetts: The MIT Press, 2011.
  2. Charles Petzold. Programming Windows / 5th Ed. – Microsoft Press, 1998.
  3. <https://www.midi.org/>
  4. <http://www.giordanobenicchi.it/midi-tech/lowmidi.htm>
  5. <http://www.muzoborudovanie.ru/articles/midi/midi1.php>
  6. <https://docs.microsoft.com/>
  7. <http://rsdn.org/article/multimedia/midi.xml>
  8. <http://faydoc.tripod.com/formats/mid.htm>
  9. <http://www.music.mcgill.ca/~ich/classes/mumt306/StandardMIDIfileformat.html>
  10. <http://www.audacity.ru/p9aa1.html>
  11. <https://en.cppreference.com/w/>

**Выполненная производственная работа**

1. Изучение теоретических материалов. Комментарий: обзор протокола MIDI, изучение спецификации MIDI-протокола; изучение функций работы с MIDI-мультимедиа через интерфейс Win32 API; изучение классических оконных приложений C++ на Win32 API; изучение функций работы с окнами Win32 API, событийно-ориентированное программирование; изучение спецификаций C++: макросы, файловые потоки, wide-char, лямбда-функции, побитовые операции, параллельное программирование (thread, mutex, lock, atomic), заголовок chrono, итераторы; формат файлов .MID, разбор и запись таких файлов.
2. Создание проекта классического оконного приложения C++ в IDE Microsoft Visual Studio. Комментарий: создание структуры проекта, сборка приложения, работа с Git, использование классического шаблона WinAPI приложения, подключение всех необходимых зависимостей.
3. Реализация функции «MIDI-клавиатура». Комментарий: создание диалогового окна, работа с MIDI-сообщениями – громкость, патчи, каналы, события; обработчик клавиш real-time.
4. Реализации функции «Проигрывание файлов». Комментарий: создание диалогового окна, парсинг (разбор) .MID файла, работа с ShellApi (для открытия отчёта), реализация параллельного проигрывания нескольких треков, реализация обработки ошибок, работа с файловыми потоками ввода/вывода в текстовом и бинарном режимах.
5. Реализации функции «Запись файла». Комментарий: создание диалогового окна, создание .MID файла по формату, более продвинутое использование элементов управления.
6. Тестирование и отладка разработанного приложения. Комментарий: разработка тестовых единиц (или же просто чеклиста) и их проверка.

**Заключение**

В ходе производственной практики я:

* получил знания о протоколе MIDI, о взаимодействии ОС Windows с подсистемой MIDI, о формате .MID файлов
* укрепил навыки проектирования и тестирования приложений
* получил новые навыки событийно-ориентированного программирования
* получил новые навыки работы с оконным интерфейсом Win32 API
* получил новые навыки работы с MIDI-устройством через интерфейс Win32 API
* укрепил имеющиеся и получил новые навыки программирования на языке C++ (в т.ч. параллельные потоки, разбор файлов и их создание, работа со временем [chrono])
* получил новые навыки работы с файлами в двоичном режиме (использование сдвигов, масок, особенности чтения и записи)

По результатам прохождения производственной практики я разработал приложение, удовлетворяющее поставленной задаче и имеющее конечную функциональность, что позволило выполнить основную цель производственной практики: получение новых навыков, новых знаний, а также укрепление навыков и знаний, полученных ранее в ходе обучения, по выбранному направлению подготовки.

**Приложение 1. Пример отчёта о загруженном .MID файле**

Filename: pirate.mid

MIDI Format: 1

Tracks number: 3

Ticks per quarter note: 192

===== TRACK #1=====

Track length: 191 bytes

Track Name: He's A Pirate

Text: By Written by Klaus Badelt

Copyright: Arrangement © 2004 Cerullean Pictures

Copyright: Original Score © Walt Disney Pictures

Text: Generated by NoteWorthy Composer

Tempo: 300000 (200bpm)

Key Signature: 255

Minor Key: 0

Time Signature: 6/8

ClocksPerTick: 24

32per24Clocks: 8

===== TRACK #2=====

Track length: 5320 bytes

Unrecognised MetaEvent: 00100001

Unrecognised Status Byte: 00000000

Track Name: Right Hand

===== TRACK #3=====

Track length: 6369 bytes

Unrecognised MetaEvent: 00100001

Unrecognised Status Byte: 00000000

Track Name: Left Hand

**Приложение 2. Листинг файлов разработанного приложения**

1. Неблокирующее работу родительского окна. Модальное окно – напротив, блокирует родительское. [↑](#footnote-ref-1)
2. BPM (Beats Per Minute) – основной показатель в музыке, определяющий скорость исполнения композиции. Стандартное распространенное значение – 120 ударов в минуту. Прямо коррелируется с MIDI-темпом, представленным в длительности одной четвертной ноты в микросекундах. [↑](#footnote-ref-2)
3. Тик (англ. tick) – единица отсчета для MIDI-устройств [↑](#footnote-ref-3)