

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ПРОГРАМИ MIDI- ПРОГРАВАЧА. DEVELOPMENT OF A
MIDI PLAYER SOFTWARE.

Виконав: студент(ка) 4 курсу, групи КНТ-218
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

Омельченко Є.В.

(прізвище та ініціали)

Керівник Сердюк С.М.

(прізвище та ініціали)

Рецензент Касьян М.М.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 122 Комп'ютерні науки
 (код і найменування)
 Освітня програма (спеціалізація) Комп'ютерні науки
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” _____ 2022 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТА(КИ)

Омельченко Євгену Владиславовичу
 (прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка програми midi-програвача. Development of a MIDI Player Software.

керівник проекту (роботи) Сердюк Сергій Микитович, к.т.н., доцент,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “27” квітня 2022 року № 102

2. Строк подання студентом проекту (роботи) 7 червня 2022 року

3. Вихідні дані до проекту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали та методи. 3. Основні рішення щодо реалізації компонентів системи. 4. Керівництво програміста. 5. Керівництво оператора. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Сердюк С.М., доцент		
Нормоконтроль	Белова А.В., асистент		

7. Дата видачі завдання 28 березня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2–3 тижні	Розділ 1
3	Розробка архітектури програми.	4 тиждень	Розділ 2
4	Розробка програми.	5–6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми.	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8–9 тижні	Розділ 5, Додатки
7	Захист роботи.	10 тиждень	

Студент(ка)

(підпис)

Омельченко Є.В.

(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Сердюк С.М.

(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
69 с., 3 табл., 11 рис., 2 дод., 22 джерела.

MIDI, MIDI ПРОГРАВАЧ, MIDI КОМАНДИ, МУЗИЧНА
КОМПОЗИЦІЯ, СИНТЕЗАТОР, JAVA.

Об'єкт дослідження – процеси синтезу звуку.

Предмет дослідження – методи побудови та відтворення MIDI композицій.

Мета роботи – розробка кросплатформної програми, що дозволяє створювати та програвати музичні композиції, що представлені у MIDI форматі.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування; мова програмування Java; інтегроване середовище розробки NetBeans IDE; персональний комп'ютер (ПК) з процесором (x64) не менше 1 ГГц під управлінням операційної системи (ОС) Windows 10 або іншої ОС.

Результати. Розроблено застосунок, що дозволяє створювати та програвати музичні композиції, що представлені у MIDI форматі.

Галузь використання – музична індустрія.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 69 pages, 11 figures, 3 tables, 2 appendixes, 22 sources.

MIDI, MIDI PLAYER, MIDI COMMANDS, MUSIC COMPOSITION, SYNTHESIZER, JAVA.

Object of study is a sound synthesis processes.

Subject of study are methods for building and playing MIDI songs.

The purpose of the work is development of a cross-platform application that allows to create and play musical compositions presented in MIDI format.

Materials, methods and technical means: object-oriented programming; programming language Java; integrated development environment NetBeans IDE; personal computer (PC) with a processor (x64) of at least 1 GHz running an operating system (OS) Windows 10 or other OS.

Results. An application has been developed that performs creation and playback of musical compositions presented in MIDI format.

The field of use is music industry.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ	9
1 Аналіз предметної області	11
1.1 Огляд сфер застосування MIDI програвачів та принципів їх роботи	11
1.1.1 Сфери застосування MIDI програвачів	11
1.1.2 Синтез звуку	14
1.2 Аналіз програмних аналогів	Ошибка! Закладка не определена.
1.3 Розробка технічного завдання	24
1.3.1 Підстава для розробки	24
1.3.2 Мета розробки та її призначення	24
1.3.3 Основні вимоги до програмного продукту	24
1.3.4 Вимоги до технічних засобів, операційної системи і мови програмування	25
1.3.5 Вимоги до інформаційної та програмної сумісності	26
1.3.6 Вимоги до програмної документації	26
2 Матеріали та методи	27
2.1 Специфікація MIDI формату	27
2.2 Використання MIDI протоколу	38
2.3 Розробка структури програми	42
3 Основні рішення щодо реалізації компонентів системи	44
3.1 Розробка алгоритму роботи програми	44
3.2 Розробка програми	45
4 Керівництво програміста	49
4.1 Призначення та умови застосування програми	49
4.2 Характеристики програми	46
4.3 Вхідні та вихідні дані	49
4.4 Повідомлення програмісту	49
5 Керівництво оператора	50

5.1 Призначення програми 50

5.2 Умови використання програми..... 50

5.3 Виконання програми 50

Висновки..... 52

Перелік джерел посилання 53

Додаток А. Текст програми 55

Додаток Б. Слайди презентації..... 63

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

DAW	– Digital Audio Workstation;
MIDI	– Musical Instrument Digital Interface;
ГГц	– гігагерц;
Мб	– мегабайт;
ОС	– операційна система;
ПК	– персональний комп'ютер;
ПЗ	– програмне забезпечення.

ВСТУП

В останнє десятиліття у музичній промисловості дедалі більше почали покладатися на використання програмних синтезаторів. Це сталося внаслідок того, що лише кількома клацаннями миші композитор може створити практично будь-який звук, який йому необхідний, з повною поліфонією та будь-яким ефектом, про який він тільки може побажати.

Синтезатори зазвичай створюють звуки, генеруючи звукові хвилі з допомогою таких методів: субтрактивний синтез; адитивний синтез та синтез частотної модуляції [1]. Отримані таким чином звуки можуть бути змінені такими компонентами як фільтри, які обрізають або підвищують частоти; конвертами, що управляють артикуляцією або тим, як ноти починаються та закінчуються; та низькочастотними генераторами, які модулюють такі параметри, як висота тону, гучність або характеристики фільтра, що впливають на тембр.

На синтезаторах зазвичай грають за допомогою клавіатури або керують секвенсорами, програмним забезпеченням або іншими інструментами. Їх також можна синхронізувати з іншим обладнанням за допомогою стандарту MIDI.

Стандарт MIDI був розроблений для того, щоб електронні або цифрові музичні інструменти могли взаємодіяти один з одним, а також щоб один інструмент міг керувати іншим. Наприклад, MIDI сумісний секвенсор може запускати ритми, що створюються звуковим модулем ударних. Аналогові синтезатори, які не мають цифрових компонентів, можуть бути модернізовані комплектами, які перетворюють MIDI повідомлення на аналогові керуючі напруги [2]. Коли відтворюється нота на MIDI інструменті, генерується цифрове MIDI повідомлення, яке можна використовувати для запуску ноти на іншому інструменті.

Можливість дистанційного керування дозволяє замінювати повнорозмірні музичні інструменти звуковими модулями меншого розміру та дозволяє музикантам комбінувати інструменти для досягнення повнішого

звучання або створювати комбінації звуків синтезованих інструментів, таких як акустичне піаніно та струнні. MIDI також дозволяє дистанційно керувати іншими параметрами інструменту, такими як гучність, ефекти тощо.

Набір відрегульованих параметрів може бути збережений у внутрішній пам'яті пристрою у вигляді патча, і ці патчі можуть бути вибрано за допомогою змін програми MIDI [3].

Отже, розробка програмного музичного MIDI програвача є актуальною практичною задачею, яка потребує свого вирішення у даній кваліфікаційній роботі бакалавра.

Результатом використання програмного застосунку, що розробляється є створені користувачем композиції та їх програвання у заданій послідовності.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд сфер застосування MIDI програвачів та принципів їх роботи

1.1.1 Сфери застосування MIDI програвачів

MIDI – це аббревіатура, що означає цифровий інтерфейс музичних інструментів. Це спосіб підключення пристроїв, які створюють звук та керують ним, таких як синтезатори, семплери та комп'ютери, щоб вони могли взаємодіяти один з одним за допомогою MIDI повідомлень. Це дозволяє одній клавіатурі запускати звуки на іншому синтезаторі або записувати музику у формі, що дозволяє легко редагувати ноти, гнучко оркеструвати та аранжувати пісні. Віртуальні інструменти – комп'ютерні програми, що імітують апаратні синтезатори та семплери, – також взаємодіють із програмним забезпеченням для секвенування, що працює на тому ж комп'ютері, за допомогою MIDI повідомлень.

MIDI дуже популярний у музичному виробництві, зокрема у формі домашньої студії звукозапису. Розглянемо переваги такого підходу:

- розмір створеної музичної композиції (MIDI файли в порівнянні з аудіофайлами мають значно менший розмір). Причина у тому, що вони зберігаються як ряд простих чисел, а не як складний аудіофайл (MP3 або WAV). З файлами меншого розміру значно зменшується обсяг роботи, яку має виконувати комп'ютерна система. Це також є перевага у випадку співпраці з іншими музикантами. Якщо необхідно надсилати файли один одному, використовувати MIDI буде набагато швидше, ніж надсилення великої кількості аудіофайлів. Але треба пам'ятати, що колеги повинні мати той самий віртуальний інструмент у своєму програмному забезпеченні цифрової робочої аудіостанції (DAW), щоб мати можливість чути те, що було створено іншим;

- можливість редагування. Завдяки тому, що створені користувачем MIDI треки просто зберігаються у його DAW у вигляді серії чисел, можна редагувати ці числа, як необхідно користувачу, причому навіть після того,

композиція вже була створена та збережена;

- бюджет. Мабуть, найголовніша перевага для музикантів, що використовують свою домашню студію, – це мати широкий спектр музичних можливостей за обмежений бюджет. Для того щоб зібрати струнний квартет або повну «живу» ударну установку необхідні значні кошти та час. Використовуючи MIDI, музикант може використовувати віртуальні інструменти та створювати повні треки з мінімальним бюджетом;

- різноманітність способів введення даних. MIDI сигнали можна надсилати через будь-які пристрої введення, які можуть бути розроблені відповідно до вимог музиканта. Частіше за все використовуються клавіатури, але під терміном «MIDI інструмент» може розумітися будь-що. Багато електронних ударних, наприклад, надсилають повідомлення через MIDI. Якщо виконавець вдаре по барабану сильніше, то буде отримано вище значення швидкості, і, як і у випадку з клавіатурою, це значення зазначається в DAW.

Звичайно, у більшості випадків там, де є переваги, є і недоліки. Все це залежить від особистого смаку, і якщо у користувача є якісні плагіни або зразки віртуальних інструментів, можливо, вони не будуть для нього недоліком.

Виділяють наступні недоліки домашніх студій [22]:

- звуки не реалістичні. Деякі музиканти стверджують, що віртуальні інструменти та семпли ніколи не звучатимуть так само, як живі. Це може бути дійсно справедливим для деяких інструментів. Але для багатьох домашніх музикантів з обмеженим бюджетом просто немає обладнання або студійних налаштувань, які відповідали б тому, що можна отримати за допомогою MIDI. Наприклад, щоб отримати якісні записи ударних, потрібно мати кімнату з відмінною акустикою, правильне налаштування мікрофонів та налаштовану ударну установку;

- MIDI не можна використовувати для передачі голосових даних. Для запису вокалу в аудіофайли зазвичай використовують вхід XLR;

- точність відтворення залежить від одержувача файлу, який повинен мати відповідний цифровий інструмент або плагіни, завантажені в його DAW,

аналогічний тому що використовував автор.

Інструменти, подібні до синтезаторів, з'явилися вперше в США в середині 20-го століття. Наприклад, такий інструмент, як RCA Mark II, керувався за допомогою перфокарт та використовував сотні електронних ламп [4]. В 1970 менші за розміром і більш дешеві синтезатори Minimoog стандартизували синтезатори як автономні інструменти з вбудованими клавіатурами.

У 1978 році компанія Sequential Circuits випустила синтезатор Prophet-5, в якому вперше використовувалися мікропроцесори, що дозволяли користувачам зберігати звуки. У 1982 році був представлений MIDI - стандартизований засіб синхронізації електронних інструментів, який залишається галузевим стандартом і зараз. Перший серійний синтезатор Yamaha DX7 було випущено в 1983, що призвело до популяризації цифрового синтезу.

Ранні синтезатори могли відтворювати лише одну ноту за раз, що робило їх придатними для басових партій та соло. З появою поліфонічних синтезаторів у 70-х і 80-х роках «клавішні у року знову почали відходити на задній план, щоб використовуватися швидше для заповнення та створення атмосфери, ніж для соло» [5].

У 1970-х роках композитори електронної музики, такі як Жан Мішель Жарр та Ісао Томіт, випустили успішні інструментальні альбоми під управлінням синтезатора. Проте деякі виконавці вважали використання синтезаторів для створення звуків «шахрайством». Наприклад, гурт Queen написав у примітках до одного зі своїх альбомів, що він їх не використовував.

У 1997 році були випущені ReBirth фірми Propellerhead Software та Reality від Seer Systems – перші програмні синтезатори, на яких можна було грати в реальному часі за допомогою MIDI. У 1999 році оновлення музичного програмного забезпечення Cubase дозволило користувачам запускати програмні інструменти (включаючи синтезатори) як модулі, що викликало хвилю нових програмних інструментів. Reason від Propellerhead, випущений у 2000 році,

представив набір віртуального студійного обладнання.

Ринок патчованих та модульних синтезаторів відновився наприкінці 1990-х років. У 2010-х роках такі компанії, як Moog, Korg, Arturia та Dave Smith Instruments, представили нові доступні аналогові синтезатори. Відновлений інтерес пояснюється привабливістю недосконалих «органічних» звуків і більш простих інтерфейсів, а також сучасною технологією поверхневого монтажу, що зробило аналогові синтезатори дешевшими та швидшими у виробництві.

Синтезатори часто використовуються в саундтреках до фільмів та телебаченні. Наприклад, синтезатори ARP використовувалися для створення звукових ефектів у науково-фантастичних фільмах 1977 року «Близькі контакти третього ступеня» [5] та «Зоряні війни». У 70-х і 80-х роках синтезатори використовувалися в музиці до трилерів і фільмів жахів, включаючи такі культові фільми як «Заводний апельсин» (1971), «Апокаліпсис сьогодні» (1979), «Туман» (1980) та «Мисливець на людей» (1986).

Композитор Бред Фідель використав синтезатор Prophet для запису саундтреку до фільму «Термінатор» (1984) [6], а режисер Джон Карпентер широко використав їх для своїх саундтреків. Синтезатори використовувалися для створення тем для телешоу, включаючи «Лицар доріг» (1982), «Твін Пікс» (1990) та «Дуже дивні справи» (2016).

В даний час синтезатор використовується майже у всіх жанрах музики і вважається одним із найважливіших інструментів у музичній індустрії. Програмні синтезатори тепер можна запускати як плагіни або вбудовувати окремі мікročіпи в будь-якому електронному пристрої. Згідно з англійським журналом Fast у 2016 році, «синтезатор так само важливий і всюдисущий у сучасній музиці, як і людський голос» [7].

1.1.2 Синтез звуку

Розглянемо існуючі методи синтезу звукових коливань, які використовуються у музичних синтезаторах. Нині найбільше застосування

знайшли такі методи [1], [8]:

- субтрактивний синтез;
- адитивний синтез;
- синтез частотної модуляції (FM).

Найбільш поширеним є субтрактивний синтез – це метод синтезу музичних тембрів, який починається з форми хвилі, що містить велику кількість гармонік, і потім фільтрується за допомогою фільтра та інших процесів (заснований на відніманні елементів один з одного). Базові форми гармонійних хвиль являють собою прості форми, такі як пилоподібні та прямокутні хвилі. Їх легко відтворити електронним способом за допомогою джерела звуку синтезатора – осцилятора. Але самі по собі (з музичного погляду) вони не становлять інтересу.

Розглянемо принцип субтрактивного синтезу та необхідний склад компонент для реалізації даного методу (рис. 1.1) [8].

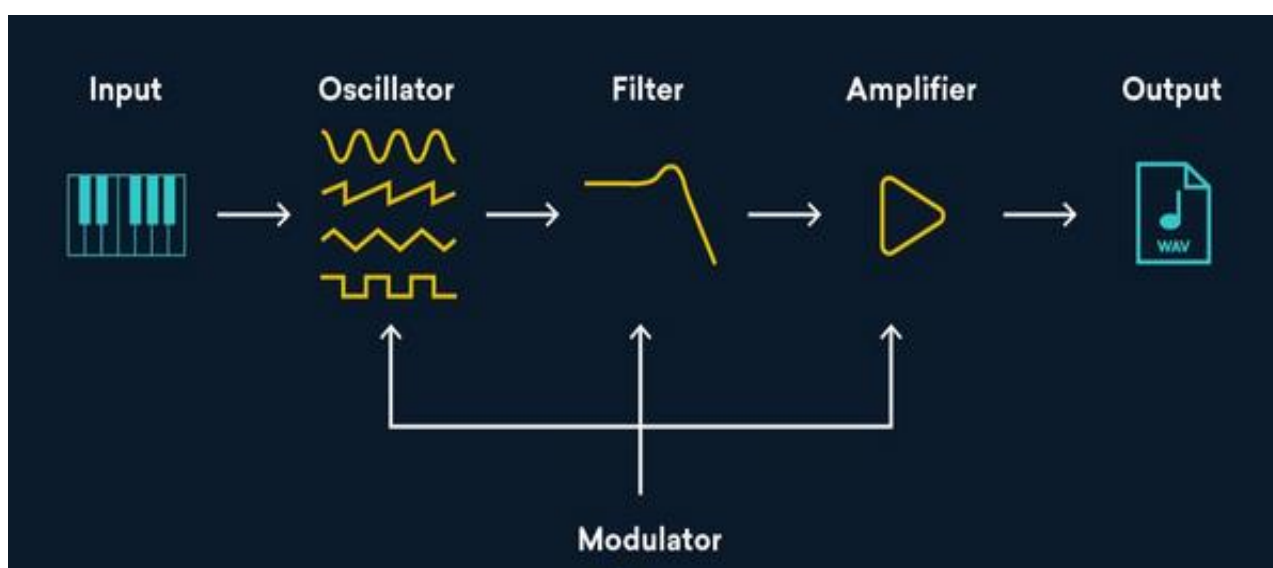


Рисунок 1.1 – Структурна схема типового субтрактивного синтезатору

Як було зазначено вище, осцилятор створює хвилю, різноманітну за формою, яка служить стартовою точкою для створюваного композитором твору (у вигляді патчу).

Іншими компонентами типового субтрактивного синтезатора є фільтр,

підсилювач, генератор огинаючої та низькочастотний генератор.

Щоб сигнал після проходження всіх частин звучав музично, необхідно сформувати його динаміку, частотний баланс і текстуру [8]. Розглянемо кожну частину сигнального ланцюжка, наведеного на рис. 1.1.

Формування звуку починається з осцилятора. Незважаючи на те, що це лише початок ланцюжка, обрані музикантом налаштування осцилятора мають великий вплив на звучання вже готового патчу.

Більшість синтезаторних генераторів пропонують вибір кілька різних хвильових форм. Найпоширенішими з них є: квадратна (меандр); пилкоподібна та трикутна. Квадратна хвиля має найнасиченіший додатковими гармоніками початковий зміст. Прямокутний генератор зберігає найвищі частоти після фільтрації. Але у прямокутних хвилях є й інші переваги – деякі прямокутні генератори мають можливість змінювати форму прямокутного візерунка. У цих осциляторах квадратна форма називається пульсовою хвилею. Установка ширини імпульсу на різні значення дає різні результати. Пилоподібний сигнал включає наступну гармоніку і має дещо інший характер. Трикутні хвилі мають найм'якше звучання.

Велика кількість субтрактивних синтезаторів мають більше одного генератора. Користувач може вибрати форму, частоту та величину розладу кожного генератора. Шляхом змішування двох осциляторів різної форми та частоти можна ускладнити свої основні форми хвилі, що створюються у субтрактивному синтезі. Тому останньою частиною секції генератора музичного синтезатора зазвичай є мікшер. Мікшер поєднує сигнали певного генератора з будь-якими іншими джерелами звуку, які можуть бути в синтезаторі. Цими джерелами можуть бути допоміжні генератори, які грають, наприклад, на октаву нижче за основний генератор, зовнішнім аудіовходом для синтезатора або джерелом шуму. Причому шум може допомогти додати присутності до атаки коротких звуків, таких як щипки та перкусія.

Наступним елементом у сигнальному ланцюжку є фільтр. Саме тут дійсно відбувається «віднімання». Багато синтезаторів пропонують різні

багаторежимні фільтри, але найпоширенішим типом фільтра субтрактивному синтезаторі є фільтр нижніх частот [8].

Цей фільтр "відсікає" високі частоти, що різко змінює звук. Кожна позиція частоти фільтра нижніх частот розкриває інший гармонійний характер генератора. Фільтрування цих високих частот різко змінює звук.

Важливою характеристикою фільтра є резонанс, що створює посилення у області частоти зрізу. Шляхом зміни частоти зрізу та резонансу фільтра, можна створити багату палітру з найвідоміших звуків у синтезі.

Наступною ланкою в ланцюжку є підсилювач. Він необхідний для доведення сигналу до рівня, достатньо сильного для підключення до іншого обладнання, такого як мікшер або аудіоінтерфейс і для керування гучністю в синтезаторі.

Для створення множини звуків використовується наступна структурна складова – генератор обвідної. Обвідні можуть бути застосовані до будь-якого структурного елементу синтезатора (рис. 1.1). Однак, найчастіше з його допомогою модулюються сигнали, що надходять на підсилювач і фільтр.

Останнім ключовим ланкою в ланцюжку субтрактивних сигналів є низькочастотний осцилятор (нижче 20 Гц), що створює модуляцію звуку синтезатора, що повторюється. Ефект від даної модуляції аналогічний ручному управлінню (обертанню ручки синтезатора вперед і назад з однаковою швидкістю щоразу – це і є низькочастотна модуляція).

Адитивний синтез – це метод синтезу звуку, що створює тембр звуку шляхом складання синусоїдальних хвиль (гармонійних складових). Тембр музичних інструментів можна розглядати з позицій теорії Фур'є, що складається з множини гармонійних або негармонічних обертонів. Кожен обертоном є синусоїдою різної частоти і амплітуди, яка наростає і згасає з часом через модуляцію сигналу генератором обвідної або низькочастотним генератором. Найчастіше при адитивному синтезі генерують звук, додаючи у вихідний сигнал кількох генераторів синусоїдальних хвиль.

Альтернативні реалізації можуть використовувати заздалегідь обчислені

хвильові таблиці або зворотне швидке перетворення Фур'є. Фактично, адитивний синтез спеціалізується на дисонансних звуках, звуках дзвіночків і «повітряних подушок» [9].

При синтезі із частотною модуляцією (FM) сигнали модулюються частотою інших сигналів. Результируючий складний сигнал може використовуватися для модуляції наступного і так далі. FM-синтез може імітувати акустичні звуки, такі як фортепіано, струнні та органи. FM-синтез характеризується як «різкий», «скляний» та «холодний» [7].

1.2 Аналіз програмних аналогів

Для розробки технічного завдання на проєктування та більш детального з'ясування вимог до програмного продукту, що розробляється проведено аналіз існуючих програмних аналогів програвачів MIDI.

Зрозуміло, що вибір оптимального застосунку залежить від конкретних завдань, що має вирішити музикант (композитор) з його допомогою. Очевидно, що ці завдання можуть змінюватися з часом. Тому при порівнянні аналогів у першу чергу будемо використовувати як економічні показники, так і ті, що обумовлюють можливості необхідні для створення нових музичних композицій та експлуатації:

- ціна;
- можливість створення MIDI файлів з набору нот;
- можливість вибору музичного інструменту для відтворення MIDI мелодії;
- можливість застосовування ефектів;
- кросплатформність.

Стандартною програмою для програвання файлів мультимедіа в операційних системах сімейства Windows є програма Windows Media Player. Це безкоштовна програма з відкритим вихідним кодом має можливість програвання більшості існуючих типів файлів, а також можливості, такі як

копіювання музики з компакт-диску на комп'ютер, записи музики з комп'ютера на компакт-диск і можливість купувати музику в інтернет-магазинах. Але недоліками є те, що ця програма не має функції створення MIDI файлів з набору нот, не має можливості вибору музичного інструменту для відтворення MIDI мелодії.

Цей проєкт в основному підтримується спільнотою на форумі Doom9. Активними відгалуженнями є Media Player Classic – Home Cinema (MPC-HC) від clsid2 [13] и Media Player Classic – Black Edition (MPC-BE) від aleksoid [14].

Інтерфейс Windows Media Player наведено на рис. 1.2.

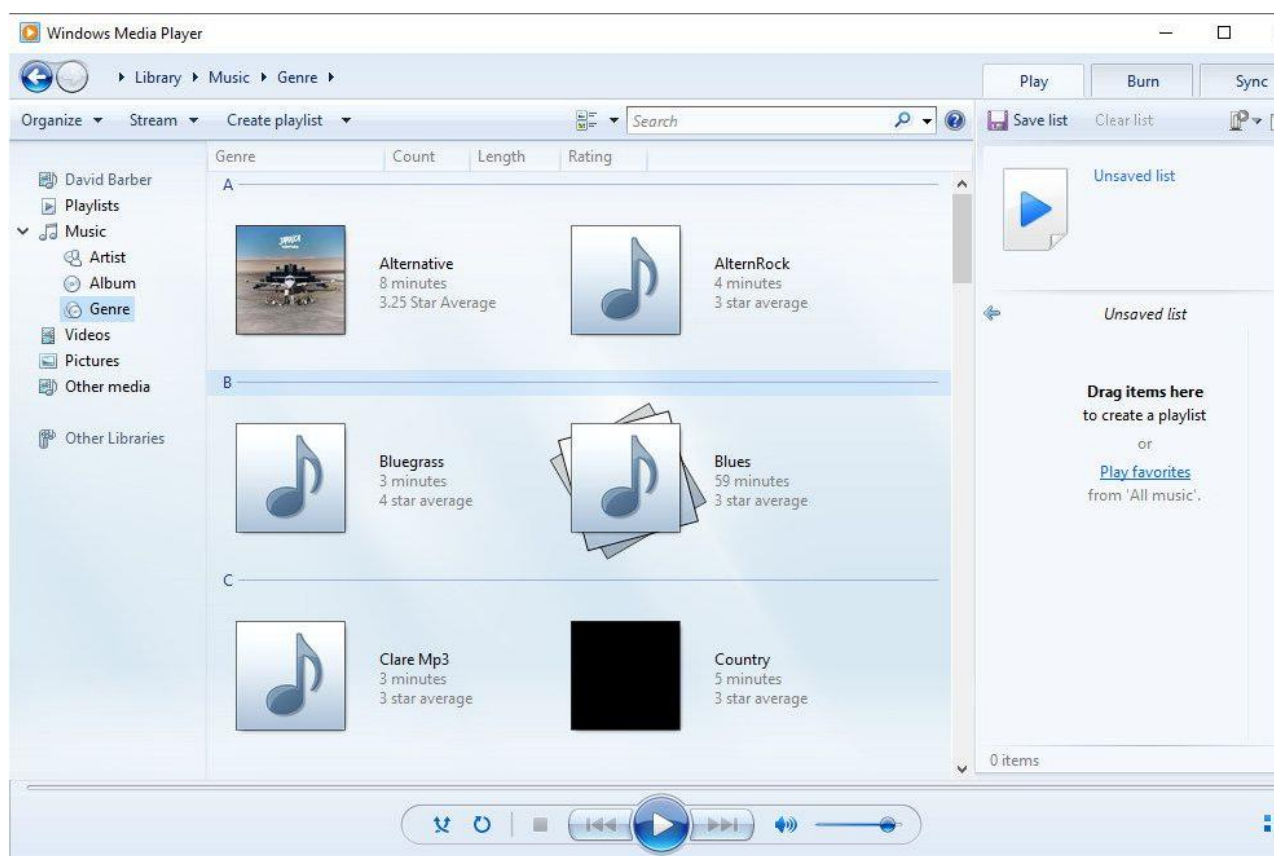


Рисунок 1.2 – Інтерфейс програми Windows Media Player

Ще одним програмним аналогом є Acid Music Studio 11 – аудіо-редактор від компанії Sony Creative Software Inc [15]. Це спеціалізована професійна програма, що дозволяє створювати музичні композиції на основі інтегрованих або власних семплів. Вона має підтримку 64-бітової архітектури, більш ніж

2500 лупів, що налаштовуються (зокрема, хіп-хоп, хаус рок та інші), вісім віртуальних інструментів та шість наборів різноманітних ефектів. Також цей застосунок дозволяє використовувати необмежену кількість аудіо-доріжок. За допомогою цієї програми можна фільтрувати імпортовані MIDI-треки, редагувати ритмічну сітку.

Але недоліками цього редактору є те, що це платна програма (коштує 59,99 €), а також вона складна у використанні, має багато професійних функцій, які не знадобляться звичайному користувачу. Інтерфейс програми Acid Music Studio 11 наведено на рис. 1.3.

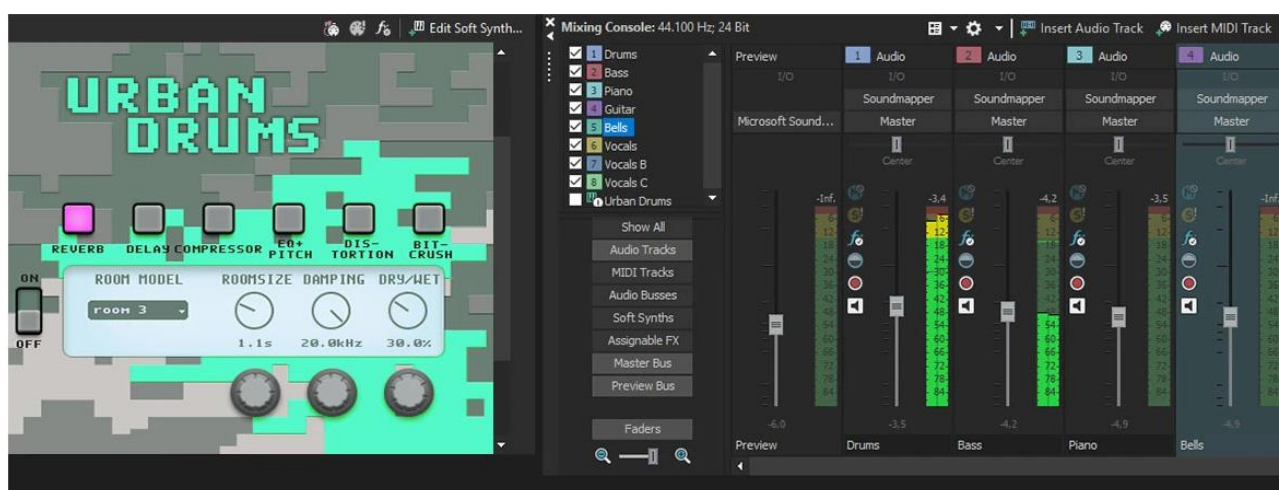


Рисунок 1.3 – Приклад інтерфейсу програми Acid Music Studio 11 [15]

Наступний аналог, це Sweet Midi Player 32 – програмний застосунок для прослуховування не тільки всіх типів MIDI файлів, але і для самостійної зміни MIDI файлів. Інтерфейс цієї програми, подібний до мікшера (рис. 1.4), дозволяє користувачу редагувати MIDI повідомлення, змінювати темп, транспонувати музику, вимкнути звук бажаних MIDI каналів та зберегти кінцевий результат на диск. За допомогою простих у використанні функцій «вирізати», «копіювати» та «вставити» можна змінити розташування файлу. Sweet Midi Player 32 також включає в себе функцію Jukebox, яка дозволяє створювати списки відтворення до 500 пісень. Крім того, Sweet Midi Player 32 має функцію відображення пісень, доступних у файлах MIDI караоке.

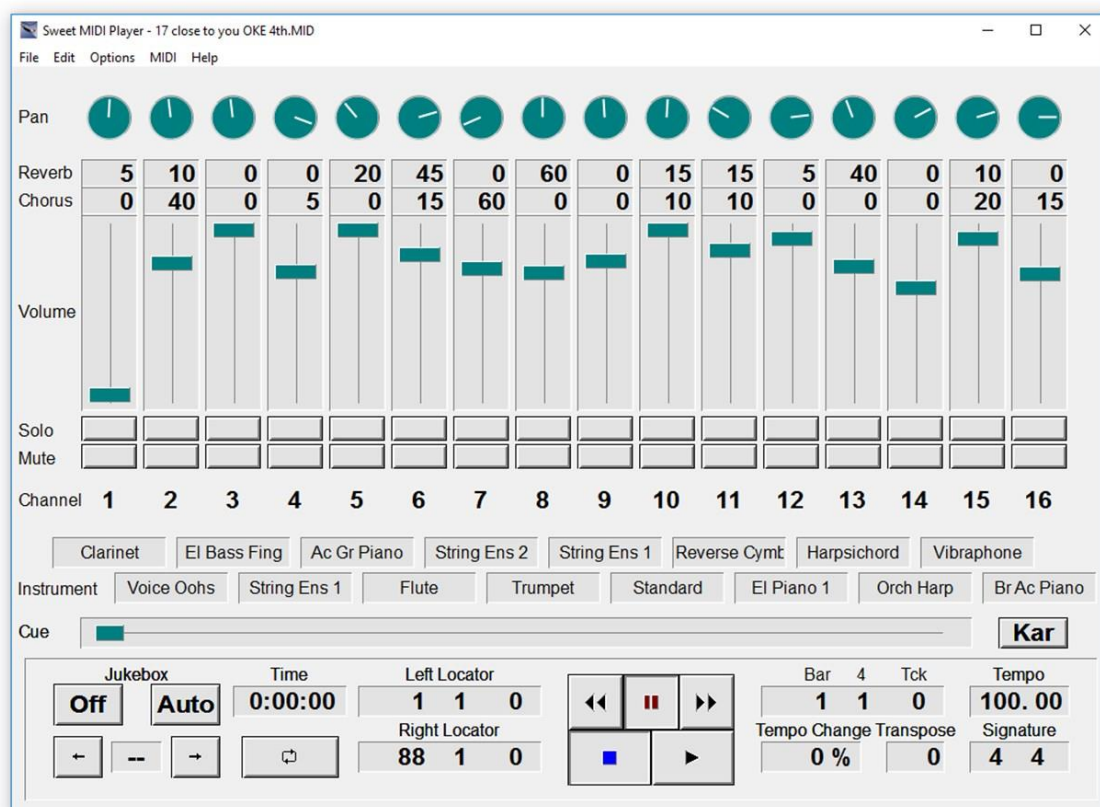


Рисунок 1.4– Приклад інтерфейсу програми Sweet Midi Player [16]

Перевагами цього ПЗ є те, що воно дозволяє зберігати редаговані MIDI композиції та дуже легке у використанні для початкових користувачів, так як багато налаштувань встановлено у потрібному форматі і вони не підлягають зміні, але водночас це є недоліком для досвідчених користувачів, бо за бажанням змінити темп, або розмір треку, їх спіткає невдача.

До недоліків також слід віднести те, що композиції вже заздалегідь сформовані, і можливо змінювати тільки готові композиції, без побудови власної. Для початкових користувачів це зручно, але для досвідчених – це утилка можливостей. Крім того, для запуску цієї програми під керуванням ОС Windows необхідно встановлювати емулятор пристрою Android (Nox App або Bluestacks). Застосунок є умовно-безкоштовним – \$29.95.

Наступним аналогом є Soundfont Midi Player 6.0 – потужний програвач і редактор MIDI файлів (рис. 1.5). Він має безліч корисних функцій з гнучкими налаштуваннями. Цей застосунок дозволяє відключати солюючі ноти або

замінювати їх іншими, регулювати басы, міняти тональність і тривалість звучання окремих нот або частин музичного файлу.

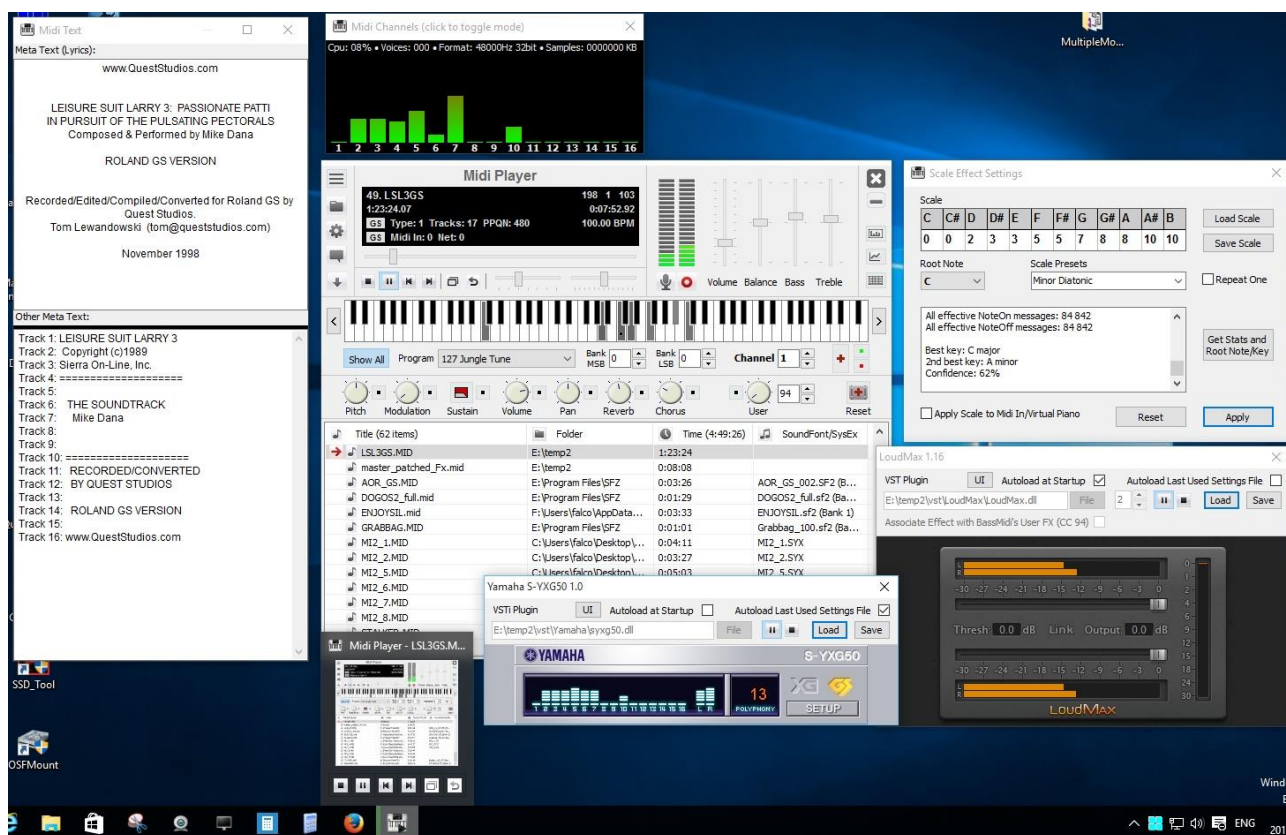


Рисунок 1.5– MIDI програвач Soundfont Midi Player 6.0 [17]

Його особливостями є [17]:

- вхідні та вихідні порти MIDI, що повністю налаштовуються;
- модифікація програми та банку в режимі реального часу на будь-якому MIDI каналі;
- ефекти в реальному часі на будь-якому MIDI каналі;
- змінні темп і висота звуку під час відтворення;
- віртуальне піаніно (функціональні клавіші F1-F8 для зміни октави);
- MIDI відправлення та отримання через мережу;
- підтримка міді файлів караоке;
- присутній аналізатор спектру;

- підтримка відтворення файлів модулів (mod,xm,s3m,it) шляхом конвертації;
- налаштування ефекту масштабу (транспонування ноти в різну гамму;
- повністю підтримує Windows 9x, XP, 7, 8, 10.

Як було описано вище це досить гнучке та корисне ПЗ, але суттєвим недоліком цього програвача є те, що в режимі реального часу відбувається програвання тільки вибраного інструменту, без можливості побудови бажаної композиції.

Крім того, у цьому застосунку відсутні наступні можливості та є обмеження:

- налаштування функцій швидкого доступу;
- потребує тільки 64-бітної архітектури;
- відсутня кросплатформність.

Результати аналізу програмних аналогів наведено у табл. 1.1.

Таблиця 1.1 – Порівняльний аналіз програмних аналогів

Параметр	Windows Media Player	Acid Music Studio 11	Sweet Midi Player	Soundfont Midi Player 6.0
1. Можливість створення MIDI файлів з набору нот	-	+	-	+
2. Можливість вибору музичного інструменту для відтворення MIDI мелодії	-	+	-	-
3. Можливість застосовування ефектів	-	+	+	+
4. Кросплатформність	-	-		-
5. Ціна	Безкоштовна	59,99 €	\$29,95	Безкоштовна

Отже, існує багато програм, які вирішують подібні задачі, проте вони не повністю відповідають поставленій меті. Найближчим аналогом програми, що розробляється є Acid Music Studio 11. Вона єдина з наведених, яка дозволяє користувачу самостійно створювати мелодію із певного набору нот та певних музичних інструментів, але є платною та працює лише під управлінням сімейства ОС Windows.

Таким чином, програмне забезпечення, що розроблюється має реалізувати позитивні сторони розглянутих аналогів, а також усунути їх недоліки.

1.3 Розробка технічного завдання

1.3.1 Підстава для розробки

Підставою для розробки служить завдання на дипломну кваліфікаційну роботу бакалавра на тему "Розробка програми MIDI програвача", затверджене наказом № 102 від 27 квітня 2022 р. по НУ «Запорізька політехніка».

1.3.2 Мета розробки та її призначення

Метою розробки є створення кросплатформного програмного забезпечення, яке призначене для створення та програвання музичних композицій, що представлені у MIDI форматі.

1.3.3 Основні вимоги до програмного продукту

Програмний продукт, що розроблюється у даній роботі має надавати користувачеві наступні можливості:

- відтворювати MIDI композицію у заданій послідовності, з функцією побудови треку (MIDI композиції) самостійно, та водночас, для простоти

користування, без використання складних налаштувань (наприклад, вибору MIDI каналів та ін.);

- виконувати управління процесом програвання шляхом подачі команд для початку відтворення, призупинення або продовження композиції;
- формування та очищення послідовності відтворення MIDI композицій.

Програма має використовувати Java Sound API, що забезпечує низькорівневу підтримку звукових операцій, таких як відтворення аудіо та записування, міксування, синтез MIDI. Java Sound API включає декілька пакетів, серед яких – `javax.sound.midi`, який і буде використовуватись програмою. Цей пакет забезпечує інтерфейси для синтезу MIDI, упорядкування, транспорту події.

Вхідними даними для ПЗ є обрані користувачем дії для управління відтворенням файлів MIDI формату, або введених даних щодо послідовності програвання бажаних інструментів.

Вихідними даними є побудований трек, або реакція програмного застосунку на відповідну управляючу дію з боку користувача.

1.3.4 Вимоги до технічних засобів, операційної системи і мови програмування

Застосунок, що розробляється має бути кросплатформним. Це означає, що він повинен працювати під управлінням операційних систем Linux, Mac та сімейства Windows.

Для забезпечення роботи застосунку під керуванням ОС Windows 10 рекомендована наступна конфігурація комп'ютера [10]:

- 64-х бітний процесор з тактовою частотою ядра понад 1 ГГц;
- 4 Гб оперативної пам'яті;
- відеокарта з 128 Мб RAM, підтримкою DX9 і WDDM;
- більше 128 Гб місця на системному розділі;

- підтримка завантаження з USB або наявність DVD-приводу;
- маніпулятори вводу (клавіатура, миша).

Для розробки кросплатформного застосунку необхідно використовувати мову програмування Java та інтегроване середовище розробки NetBeans IDE.

1.3.5 Вимоги до інформаційної та програмної сумісності

ПЗ, що розроблюється повинно бути сумісним як з ОС Windows 10 так і з Unix-подібними ОС.

1.3.6 Вимоги до програмної документації

Програмна документація має включати: керівництво програміста, керівництво оператора, опис програми та тексти основних програмних модулів.

2 МАТЕРІАЛИ ТА МЕТОДИ

2.1 Специфікація MIDI формату

Як було зазначено у п. 1.1, MIDI – означає цифровий інтерфейс музичних інструментів. Таким чином через посередництво MIDI не відправляються ніякі аналогові звукові сигнали – тільки послідовність двійкових цифр.

Кожен прилад (або комп'ютер) розуміє і потім реагує на ці послідовності, які поєднуються потім у 8-бітові повідомлення, що підтримують швидкість передачі даних до 31250 біт в секунду.

Ці повідомлення можуть надсилати корисну інформацію, таку як:

- нота, що натиснута;
- момент, коли нота натиснута та відпущена;
- швидкість (наскільки сильно натиснута нота);
- післяторкання (при зміні натискання клавіші);
- вібрато;
- модуляція звуку.

Протокол MIDI може підтримувати до 128 нот, від ноти «До», що на п'ять октав нижче так званої ноти "середньої До" (До першої октави – C4), до ноти «Соль» на десять октав вище. Отже, це практично будь-яка нота, яку музикант коли-небудь хотів зіграти.

Інші значення, такі як швидкість, записуються як числа від 0 до 127. Де 0 означає відсутність звуку, а 127 найгучніший звук. Ці стандартизовані числа можуть бути прочитані будь-яким інструментом чи машиною, здатною розуміти MIDI. Ось чому MIDI є таким потужним інструментом створення музики.

Тип MIDI, зазначений вище, відомий як "загальний MIDI" (або стандарт General Midi (GM)), але варто пам'ятати, що його можливості залежать від віку інструменту, що використовується.

Стандарт GM Level 1 – розроблений у 1991 році і складається як

мінімум із 128 патчів (звуків інструментів або "програм"), 24 нот поліфонії, 16 каналів та мультитембральності з 16 частин.

Стандарт GM 2 – реалізований у 1999 році, включає більше звуків та функцій. Також було додано 32 ноти поліфонії та 16 одночасних звукових патчів інструментів, ефекти реверберації та хорусу.

Цифровий сигнал, можна використовувати не тільки для музики, а і для багатьох інших функцій, наприклад, для запуску світлових шоу у театральних постановках. По суті, його можна використовувати для керування будь-яким цифровим пристроєм, який може його зчитувати та обробляти.

У своїй найпростішій формі синтезатор MIDI перетворює нотні повідомлення MIDI в аудіосигнал. Більш складні синтезатори забезпечують точне налаштування результуючого звуку, дозволяючи імітувати інструменти.

Важливим моментом є те, що формат MIDI визначає не сам звук, а лише послідовність інструкцій для створення звуку у цільовому синтезаторі.

Повідомлення MIDI надсилаються у вигляді часової послідовності з одного або кількох байтів (8 біт). Перший байт – це байт STATUS, за яким часто йдуть байти DATA з додатковими параметрами. У байті STATUS біт 7 встановлено в 1, а в байті DATA біт 7 встановлено в 0. Байт STATUS визначає тип повідомлення. Кількість наступних байтів DATA залежить від типу повідомлення [10].

За винятком деяких системних MIDI повідомлень, байт STATUS містить номер MIDI каналу. Існує 16 можливих MIDI каналів, пронумерованих від 0 до 15 у шістнадцятковому форматі.

В одному і тому самому MIDI кабелі можна використовувати до 16 MID каналів для управління 16 різними інструментами, що грають незалежно один від одного.

Основними повідомленнями є NOTE ON і NOTE OFF. Повідомлення NOTE ON надсилається, коли виконавець натискає клавішу на музичній клавіатурі. Воно містить параметри для вказівки висоти тону ноти та швидкості (інтенсивності ноти при натисканні). Коли синтезатор отримує це

повідомлення, він починає грати цю ноту з правильною висотою тону та рівнем сили. При отриманні повідомлення NOTE OFF відповідна нота вимикається синтезатором.

Повідомлення NOTE ON має таку структуру:

- байт стану: 1001 SSSS;
- байт даних 1: 0TTT TTTT;
- байт даних 2: 0VVV VVVV.

Тут використовуються наступні позначення: «SSSS» – MIDI канал (від 0 до 15); "TTT TTTT" – значення висоти тону (від 0 до 127) і "VVV VVVV" – значення відносної гучності (інтенсивності) звуків, що виконуються (від 0 до 127).

Значення висоти тону визначає частоту ноти, що відтворюється (в діапазоні від 0 до 127). Позначення абсолютного числа октав засновані на широко використовуваному значенні "середнє ДО". Відповідно до табл. 2.1 ця нота має значення 60 [11].

Таблиця 2.1 – Номери нот

№ октави	Назва октави	C до	C#	D ре	D#	E мі	F фа	F#	G соль	G#	A ля	A#	B сі
-1	–	0	1	2	3	4	5	6	7	8	9	10	11
0	Субконтроктава	12	12	14	15	16	17	18	19	20	21	22	23
1	Контроктава	24	25	26	27	28	29	30	31	32	33	34	35
2	Велика	36	37	38	39	40	41	42	43	44	45	46	47
3	Мала	48	49	50	51	52	53	54	55	56	57	58	59
4	Перша	60	61	62	63	64	65	66	67	68	69	70	71
5	Друга	72	73	74	75	76	77	78	79	80	81	82	83
6	Третя	84	85	86	87	88	89	90	91	92	93	94	95
7	Четверта	96	97	98	99	100	101	102	103	104	105	106	107
8	П'ята	108	109	110	111	112	113	114	115	116	117	118	119
9	–	120	121	122	123	124	125	126	127	128	129	130	131

Щоб транспонувати ноту на одну октаву вище, необхідно додати 12 до її висоти тону. Таким чином, використовуючи MIDI, транспонування досить просте, оскільки воно виконується простим додаванням або відніманням фіксованого значення.

Значення відносної гучності зазвичай змінюється від 1 до 127, охоплюючи діапазон практично від нечутної ноти до максимального рівня ноти. Це в основному відповідає шкалі динаміки, що зустрічаються в нотному записі, в такий спосіб. Наприклад, значенню "Тихо" (Piano) відповідає 42, а значенню "Гучно" (Forte) відповідає цифра 80.

У базових синтезаторах значення відносної гучності використовується лише для визначення сили, з якою програється нота, і єдиним ефектом є те, що нота стає гучніше чи тихіше.

У складнішому синтезаторі це значення також впливатиме на якість звуку. Наприклад, на справжньому фортепіано сильніше натискання на ноту вплине не тільки на її гучність, але і на якість самого звуку, на тембр. Це відноситься практично до будь-якого реального інструменту.

Існує особливий випадок, коли відносна гучність дорівнює нулю. Повідомлення NOTE ON має те значення, що й повідомлення NOTE OFF, що відключає ноту.

Повідомлення NOTE OFF структуровано таким чином:

- байт стану: 1000 SSSS;
- байт даних 1: 0TTT TTTT;
- байт даних 2: 0VVV VVVV.

Тут, позначення SSSS та TTTT мають те ж саме значення, що й вище. VVVVVVVV – це швидкість вивільнення, яка використовується дуже рідко. За замовчуванням його рекомендують встановлювати рівним нулю.

Коли на синтезатор надсилається повідомлення NOTE ON, починає відтворюватися дана нота. Щоб почути акорд, необхідно надсилати інші повідомлення NOTE ON з різною висотою нот. При цьому, музикантам необхідно відстежувати ноти, що відтворюються, щоб вони могли надіслати

відповідне повідомлення NOTE OFF для кожної ноти, інакше ноти будуть назавжди «зависати».

Протокол MIDI обробляє до 16 різних каналів MIDI. Кожен канал має свій власний статус, наприклад, певний поточний інструмент, ноти, що відтворюються в даний момент, а також інші значення, такі як гучність, панорама і т. д.

Існує MIDI повідомлення для вказівки інструменту з визначеного списку. За замовчуванням синтезатор використовуватиме як інструмент піаніно.

Теоретично у кожного синтезатора може бути свій власний список інструментів, але стандарт General Midi визначає список з 128 інструментів, що спрощує сумісність. Більшість синтезаторів мають як мінімум режим сумісності зі стандартом GM.

За стандартом MIDI існує числове позначення для музичних інструментів. Приклад назв деяких інструментів та відповідних їм числових кодів наведено у табл. 2.2.

Таблиця 2.2 – Числові коди деяких музичних інструментів

Група	Номер	Оригінальна назва	Переклад
1	2	3	4
Piano (Фортепіано)	1	Acoustic piano	Акустичний рояль
	3	Grand piano	Великий рояль
	7	Harpsichord	Клавесин
Chromatic Percussion (Хроматичні ударні)	9	Celesta	Челеста
	10	Glockenspiel	Дзвіночки
	14	Xylophone	Ксилофон
Organ (Орган)	17	Hammond	Орган Хаммонда
	22	Accordion	Акордеон
	23	Harmonica	Гармоніка
Guitar (Гітара)	25	Nylon Guitar	Акустична гітара (з нейлоновими струнами)

Продовження табл. 2.2

1	2	3	4
	26	Steel Guitar	Акустична гітара (з металевими струнами)
	27	Jazz Guitar	Джазова електрогітара
	28	Clean Guitar	Електрогітара чиста
Bass (Бас)	33	Acoustic Bass	Акустичний бас
	34	Finger Bass	Пальцевий бас
	35	Piked Bass	Шипковий бас
Strings (Струнні)	41	Violin	Скрипка
	42	Viola	Альт
	43	Cello	Віолончель
	44	Double Bass	Контрабас
Brass (Мідні)	57	Trumpet	Труба
	58	Trombone	Тромбон
	59	Tuba	Туба
Ethnic (Народні інструменти)	105	Sitar	Ситара
	106	Banjo	Банджо
	109	Kalimba	Калімба
	110	Bagpipe	Волинка

Використовуючи різні MIDI канали, музикант може визначити для кожного певний інструмент. Після надсилання ноти на відповідні MIDI канали, ці ноти будуть відтворюватися заданими інструментами.

MIDI повідомлення, яке використовується для вказівки інструмента, називається повідомленням про зміну програми. Воно має один байт STATUS та один байт DATA:

- байт стану: 1100 SSSS;
- байт даних 1: 0NNN NNNN.

Тут NNNNNNN – номер інструменту від 0 до 127. Наприклад, трубний інструмент має номер 57 у списку GM. Ноти, що надсилаються в синтезатор після цього повідомлення, будуть відтворюватися як звук труби.

Синтезатор може містити один або кілька звукових банків, кожен з яких містить 128 звуків. Для використання певного звукового банку синтезатора необхідно спочатку активувати новий банк, а потім відправити повідомлення про зміну програми. Повідомлення вибору звукового банку (молодший байт

LSB і старший байт MSB) використовуються на додаток до повідомлення про зміну програми.

Наприклад, для використання звуку номер 3 (закодований 2) на каналі 1 (закодований 0) у банку зі специфікацією LSB=1; MSB=5, необхідно надіслати наступну послідовність MIDI повідомлень:

0xB0 - 0x00 - 0x05 (вибір звукового банку MSB);

0xB0 - 0x20 - 0x01 (вибір звукового банку LSB);

0xC0 - 0x02 (вибір звуку у поточному звуковому банку).

Після отримання цих повідомлень синтезатор відтворюватиме наступні ноти з цим конкретним звуком.

Оскільки MSB і LSB можуть варіюватися від 0 до 127, можна вибрати $128 \times 128 = 16384$ можливих звукових банків. Насправді реалізуються лише деякі з них.

Розглянемо необхідний для відтворення такту, представленого на рис. 2.1, приклад послідовності MIDI повідомлень (рис. 2.2).



Рисунок 2.1 – Приклад музичного такту [12]

Часова послідовність MIDI повідомлень, які необхідно відправити на синтезатор, щоб він відтворив цей такт на каналі 1 (він закодований як 0) з відносною гучністю 64 (меццо форте) у шістнадцятковому форматі представлена на рис. 2.2 [12].

$t=0 : 0x90 - 0x40 - 0x40$ (Початок ноти E3, крок = 64)
 $t=0 : 0x90 - 0x43 - 0x40$ (Початок ноти G3, крок = 67)
 $t=1 : 0x80 - 0x43 - 0x00$ (Кінець ноти G3, висота тона = 67)
 $t=1 : 0x90 - 0x45 - 0x40$ (Початок ноти A3, крок = 69)
 $t=2 : 0x80 - 0x45 - 0x00$ (Кінець ноти A3, крок = 69)
 $t=2 : 0x80 - 0x40 - 0x00$ (Кінець ноти E3, висота тону = 64)
 $t=2 : 0x90 - 0x3C - 0x40$ (Початок ноти C3, крок = 60)
 $t=2 : 0x90 - 0x47 - 0x40$ (Початок ноти B3, крок = 71)
 $t=3 : 0x80 - 0x47 - 0x00$ (Кінець ноти B3, крок = 71)
 $t=3 : 0x90 - 0x48 - 0x40$ (Початок ноти C4, висота тону = 72)
 $t=4 : 0x80 - 0x48 - 0x00$ (Кінець ноти C4, крок = 72)
 $t=4 : 0x80 - 0x3C - 0x40$ (Кінець ноти C3, крок = 60)

Рисунок 2.2 – Часова послідовність MIDI повідомлень

На рис. 2.2 змінна t є час у секундах. Партитура відтворюється зі швидкістю 60 ударів за хвилину, тому кожна четверта нота становить 1 секунду.

Барабанні інструменти є особливим випадком, тому що вони не мають певної висоти тону, як у фортепіано або труби. Тому для передачі відтворення ударних інструментів використовується спеціальний MIDI канал. У General Midi їм виділено канал 10. Проте, є чимало різних синтезаторів, які можна запрограмувати на прийом ударних на будь-якому каналі.

У випадку каналу, призначеного для ударних інструментів (а також для спеціальних звукових ефектів, які часто включаються в синтезатори), інформація повідомлень NOTE ON і NOTE OFF для висоти тону фактично використовується для вибору ударних або звукових ефектів, які будуть відтворюватися.

Наприклад, щоб зіграти на бас-барабані на каналі 10, необхідно спочатку сформувати і відправити наступне повідомлення NOTE ON:

0x99 0x23 0x40

У цьому повідомленні 0x99 – це байт STATUS для повідомлення NOTE ON, що використовує канал 10 (код 9). 0x23 – це десяткове число 35, яке є номером ноти, що використовується для акустичного бас-барабану у відповідності до списку GM. 0x40 – це десяткове число 64 для відносної гучності, близької до динамічного відтінку меццо-форте.

Потім для завершення відтворення необхідно надіслати таке повідомлення NOTE OFF:

0x89 0x23 0x00

На сумісні з MIDI електронні чи цифрові пристрої інформація передається від спеціалізованого програмного чи апаратного забезпечення – MIDI контролера. MIDI контролери не можуть відтворювати звуки. Вони призначені для створення потоку даних після фізичних дій музиканта (натискання на клавіші, кнопки або педалі пристрою). Одним із найпоширеніших контролерів є MIDI клавіатура.

Визначено 128 MIDI контролерів, але на практиці використовуються лише деякі з них. Метою MIDI контролера є встановлення значення параметра синтезатора, що відтворює ноти, наприклад, гучності, панорами (положення в просторі зліва направо в стерео), рівня реверберації і т.д.

Повідомлення контролера формується так:

- байт стану: 1011 SSSS;
- байт даних 1: 0CCC CCCC;
- байт даних 2: 0VVV VVVV.

Тут CCCC – номер контролера (від 0 до 127), а VVVVVVVV – значення, яке присвоєно контролеру (також від 0 до 127).

Найбільш поширені номери контролерів наступні:

- 0 = вибір звукового банку (MSB);
- 1 = колесо модуляції, що часто призначається для ефекту вібрато або тремоло;
- 7 = рівень гучності інструменту;
- 10 = панорамний (0 = ліворуч; 64 = по центру; 127 = праворуч);

- 11 = Expression (використовується також для регулювання гучності, залежно від синтезатора);
- 32 = вибір звукового банку (LSB);
- 64 = педаль Sustain (0 = педаль відсутня; ≥ 64 – педаль включена);
- 121 = всі контролери вимкнені (це повідомлення скидає всі значення контролерів для цього каналу, повертаючи значення за замовчанням);
- 123 = усі ноти відключені (це повідомлення зупиняє відтворення всіх нот, що програватимуться на даний момент).

Розглянемо як можна встановити гучність і відносну гучність. Для встановлення гучності, наприклад, на значення 100 (у діапазоні значень від 0 до 127, $100 = 0x64$) для інструменту, що грає на каналі 1 (кодується 0), необхідно надіслати таке повідомлення:

0xB0 - 0x07 - 0x64

Зміна гучності впливає на всі ноти, що відтворюються в даний момент, а також на ноти, які відтворюватимуться пізніше. Синтезатор зберігає цей рівень гучності, доки не буде надіслано інший рівень гучності.

Відносна гучність ноти передається разом із самим повідомленням NOTE ON. Вона може бути змінена після того, як нота була розпочата. Для цього можна використати регулятор гучності.

Для того щоб створити кресендо, необхідно відправити послідовність зростаючих значень гучності.

Для зміни висоти звуку нот, що відтворюються на поточному MIDI каналі, використовується наступне повідомлення Pitch Bend:

- байт стану: 1110 SSSS;
- байт даних 1: 0LLL LLLL;
- байт даних 2: 0MMM MMMM.

Тут LLLLLLL – молодший біт значення зміни висоти тону, а MMMMMMM – старший біт. 14-бітове значення зміни висоти тону визначено таким чином, що значення $0x2000$ є центром, який відповідає нормальній висоті ноти (без зміни висоти тону). Використання чисел вище $0x2000$ (до $0x3FFF$)

збільшує висоту тону, а використання чисел нижче (до 0x0000) зменшує висоту тону. Діапазон зміни висоти тону симетричний (вгору та вниз) і часто може регулюватися в самому синтезаторі. Найбільш поширене значення діапазону становить +/- 2 півтони від стандартної висоти ноти.

Зміна висоти тону часто використовується для створення ефектів глісандо та гітарного бенду. Щоб досягти цього, потрібно надсилати безперервну послідовність повідомлень про зміну висоти тону, щоб змінювати висоту тону в реальному часі. Виконувати це слід досить часто, щоб вухо не чуло надто багато кроків на кривій.

Наприклад, щоб відтворити ноту на півтону вище, необхідно надіслати значення 0x3000, у такому вигляді 0xE0 - 0x00 - 0x60.

Це означає, що значення 0x3000 має бути розділене на два 7-бітові значення, даючи 0x60 і 0x00 для частин MSB і LSB.

Існує чотири способи скинути всі ноти, що відтворюються:

- використання MIDI контролера 123. У разі відправки MIDI контролера 123 на один із MIDI каналів, синтезатор зупинить відтворення всіх нот на цьому каналі. Щоб скинути всі MIDI канали, необхідно надіслати це повідомлення кожному каналу;

- повідомлення скидання MIDI. Це повідомлення з одним байтом стану 0xFF без байтів даних. Воно має скинути синтезатор до значення за замовчуванням під час увімкнення живлення, щоб він також зупиняв відтворення всіх нот. Слід врахувати, що це повідомлення скидає весь синтезатор, а не тільки ноти, що відтворюються;

- повідомлення MIDI NOTE OFF. Для кожного каналу (від 0 до 15) та кожної висоти ноти (від 0 до 127) можна надіслати одне повідомлення MIDI NOTE OFF. Це комплексне рішення, яке вимагає надсилання великої кількості MIDI повідомлень, на обробку яких знадобиться певний час (залежно від апаратної системи MIDI, що використовується);

- повідомлення MIDI NOTE OFF. У цьому випадку використовується таблиця для відстеження повідомлень NOTE ON та OFF, які користувач

надсилає кожному каналу. Буфер із 128 байтів для кожного каналу, що представляє кількість повідомлень NOTE ON, надісланих на канал, повинен збільшуватися на NOTE ON та зменшуватися на NOTE OFF. Потім, коли слід скинути кожну ноту, необхідно переглянути таблицю та надіслати NOTE OFF для кожної ноти, яка все ще грає.

2.2 Використання MIDI протоколу

Існують різноманітні фреймворки та технології для здійснення взаємодії з пристроями MIDI, обробки звуку в реальному часі тощо. Наприклад, платформи Apple надають Core MIDI API для зв'язку з пристроями MIDI, включаючи апаратні клавіатури та синтезатори [18]. Цей інтерфейс забезпечує підключення пристроїв iOS за допомогою док-конектора або мережі. У разі використання багатоканального звуку з просторовим рендерингом або обробки звуку в реальному часі використовуються двигун AVAudioEngine та різні розширення технології Audio Unit Extensions. Ця технологія дозволяє створювати DSP-модулі, які перетворюють або створюють аудіо-сигнали. Також можуть використовуватися сигнали з інших програм, які підтримують цю функціональність. Наприклад, можна створити свій синтезатор, драм-машину або звуковий ефект, який можна підключити до віртуальної музичної студії Garage Band.

Такий фреймворк як Audio Toolbox для MATLAB надає інструменти для обробки звуку, аналізу мовлення та акустичних вимірювань. Він включає в себе алгоритми обробки аудіосигналів, таких як вирівнювання і розтягнення часу, оцінку показників акустичного сигналу, таких як гучність і різкість, а також виділення звукових функцій, таких як MFCC і тон. Програми Toolbox підтримують тестування алгоритмів у реальному часі, вимірювання імпульсної характеристики та маркування сигналів. Набір інструментів забезпечує потокові інтерфейси для ASIO, CoreAudio та інших звукових карт; MIDI пристроїв, а також інструменти для створення та розміщення плагінів VST

(ефектів) і Audio Units (плагіни, які можна використовувати для обробки звуку) [19].

За допомогою Audio Toolbox можливо прототипувати алгоритми обробки звуку в режимі реального часу або виконувати спеціальні акустичні вимірювання, передаючи аудіо з низькою затримкою на звукову карту та з неї. Також надається можливість перевірити свій власний алгоритм, перетворивши його на аудіоплагін для запуску в зовнішніх хост-програмах, таких як цифрові аудіо робочі станції. Хостинг плагінів дозволяє використовувати зовнішні аудіоплагіни як звичайні об'єкти MATLAB.

Розглянемо використання фреймворку Audio Toolbox для організації взаємодії з пристроями MIDI. Пристрої дозволяють надсилати та отримувати всі MIDI повідомлення, що описані в протоколі MIDI. MIDI – це технічний стандарт для зв'язку між електронними інструментами, комп'ютерами та супутніми пристроями. MIDI передає повідомлення про події, характерні для аудіосигналів, таких як висота тону та швидкість, а також сигнали керування параметрами та тактовими сигналами для синхронізації темпу.

MIDI пристрої мають порти входу, вихідні порти або обидва. Протокол MIDI визначає повідомлення як односпрямовані. MIDI пристрій може бути реальним або віртуальним.

Audio Toolbox дозволяє створити інтерфейс до MIDI пристрою за допомогою функції `mididevice`. Щоб створити MIDI інтерфейс для певного пристрою, необхідно використовувати функцію `mididevinfo` для запиту у системі користувача щодо доступних пристроїв. Потім створюється об'єкт `mididevice`, шляхом вказівки MIDI пристрою за ім'ям або ідентифікатором [20]:

```
mididevinfo
```

```
MIDI devices available:
```

ID	Direction	Interface	Name
0	output	MMSystem	'Microsoft MIDI Mapper'
1	input	MMSystem	'USB MIDI Interface '

```
2    output    MMSystem    'Microsoft GS Wavetable
Synth'
```

```
3    output    MMSystem    'USB MIDI Interface '
device = mididevice('USB MIDI Interface ')
device =
```

```
mididevice connected to
```

```
Input: 'USB MIDI Interface ' (1)
```

```
Output: 'USB MIDI Interface ' (3)
```

Користувач може вказати об'єкт `mididevice` для прослуховування вхідних повідомлень, надсилання вихідних повідомлень або їх обох. У наведеному вище прикладі об'єкт `mididevice` отримує MIDI повідомлення через вхідний порт під назвою «USB MIDI Interface» і надсилає MIDI повідомлення з вихідного порту «USB MIDI Interface».

Розглянемо процес конвертації MIDI повідомлень у звукові коливання.

Інформація щодо MIDI упаковується як об'єкт `NoteOn` або `NoteOff` повідомлень `midimsg` в `Audio Toolbox`. Об'єкти `midimsg` `NoteOn` і `NoteOff` мають властивості `Note` і `Velocity`:

- властивість `Velocity` вказує, наскільки сильно має програватися нота. За замовчанням, повідомлення `NoteOn` із нульовою швидкістю представляють повідомлення про відключення. Представлення повідомлень за допомогою приміток до повідомлень ефективніше, якщо використовується «Статус виконання» (`Running Status`);

- властивість `Note` вказує частоту звукового сигналу. Вона може приймати значення від нуля до 127 включно. Протокол MIDI визначає, що 60 є так звана нота «середнє До» (див. табл. 2.1).

Приклад завдання цих властивостей:

```
channel = 1;
note = 60;
velocity = 64;
```

```
msg = midimsg('NoteOn', channel, note, velocity)
```

```
msg =
```

```
MIDI message:
```

```
NoteOn Channel: 1 Note: 60 Velocity: 64 Timestamp
0 [90 3C 40]
```

Щоб інтерпретувати властивість ноти як частоту, найбільш часто використовується рівномірно темперований стрій та міжнародний стандарт частоти ноти «Ля» першої октави, що дорівнює частоті 440 Гц.

Наприклад, якщо задати:

```
frequency = 440 * 2^((msg.Note-69)/12)
```

то значення частоти дорівнює

```
frequency = 261.6256 Гц
```

Щоб отримувати та синтезувати нотні повідомлення в режимі реального часу, необхідно створити інтерфейс до MIDI пристрою. Для цього використовується функція `simplesynth`:

- отримує повідомлення MIDI ноти від зазначеного MIDI пристрою;
- синтезує звуковий сигнал;
- відтворює ноти на аудіопристрої користувача в режимі реального часу.

Для того щоб запитати назву пристрою у системі користувача, необхідно використовувати функцію `mididevinfo`. А для того, щоб прослухати вибраний пристрій, викликається функція `simplesynth` з назвою пристрою.

Розглянемо приклад використання пристрою M-Audio KeyRig 25, який реєструється з ім'ям пристрою USB 02 на комп'ютері:

```
mididevinfo
```

```
MIDI devices available:
```

ID	Direction	Interface	Name
0	output	MMSystem	'Microsoft MIDI Mapper'
1	input	MMSystem	'USB MIDI Interface '
2	input	MMSystem	'USB 02'
3	output	MMSystem	'Microsoft GS Wavetable Synth'

```

4   output      MMSystem  'USB MIDI Interface '
5   output      MMSystem  'USB O2 '

```

Після цього викликається функція `simplesynth` з назвою пристрою користувача. Ця функція `simplesynth` нотні повідомлення та відтворює їх на аудіопристрої користувача за замовчуванням.

2.3 Розробка структури програми

Виходячи з поставлених функціональних вимог, наведених у ТЗ (див. п. 1.3), роботу програми можна розділити на три окремі функціональні блоки:

- робота з інструментами;
- робота з композицією;
- робота з послідовністю.

Функціональна структура роботи програми наведена на рис. 2.3

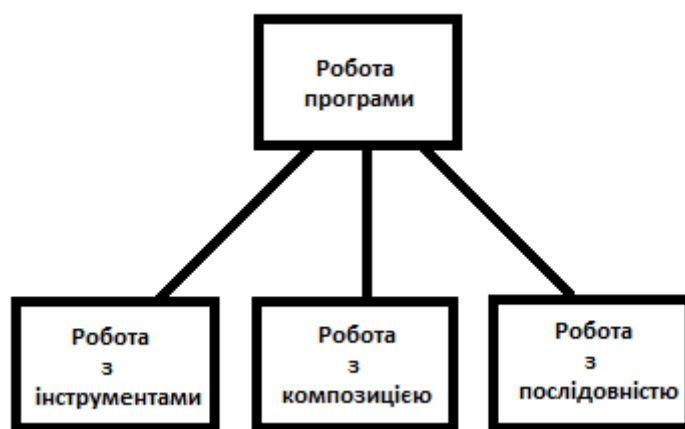


Рисунок 2.3– Функціональна структура програми

Конкретизуємо роботу кожного з цих функціональних блоків декомпозуючи їх на підзадачі.

До підзадач функціонального блоку «Робота з інструментами» відноситься задача відтворення (програвання) вибраного інструменту;

До підзадач функціонального блоку «Робота з композицією» відносяться наступні задачі:

- формування композиції для програвання у заданій користувачем послідовності;
- відтворення (програвання) сформованої композиції.

До підзадач функціонального блоку «Робота з послідовністю» відноситься задача формування послідовності з обраних інструментів.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Розробка алгоритму роботи програми

Виходячи з перелічених у п. 1.3 вимог до функціональності програми було розроблено наступний алгоритм її роботи:

- вибір послідовності інструментів;
- відтворення (програвання) MIDI композиції;
- призупинення композиції (залежить від вибору користувача);
- продовження композиції (залежить від вибору користувача);
- очищення послідовності (залежить від вибору користувача).

Даний алгоритм наведений на рис. 3.1.

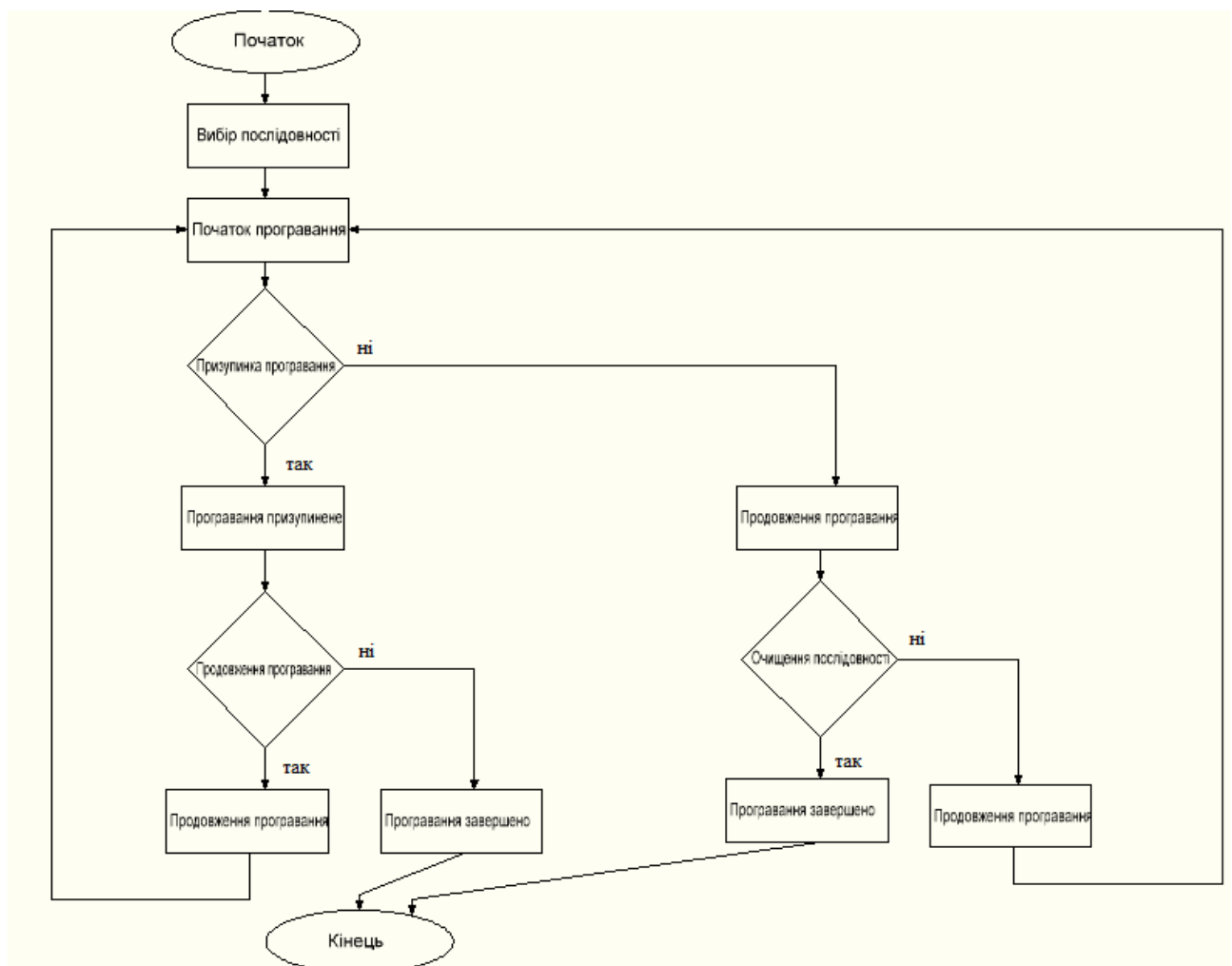


Рисунок 3.1 – Блок-схема алгоритму роботи програми

3.2 Розробка програми

Виходячи з функціональної структури програми, наведеної на рис. 2.1 було розроблено наступні класи системи:

- клас `MainClass` – головний клас для запуску застосунку;
- клас `MyNumberBeats` – містить константи, необхідні для роботи програми (довжина послідовності);
- клас `Interface` – відповідає за відображення програми (графічний інтерфейс);
- клас `MyTools` – містить інформацію про кожний інструмент;
- клас `MyMidiClass` – створення та відтворення композицій;
- клас `About` – працює з усіма типами звукової інформації, зібраної від користувача.

Розглянемо розроблені класи більш детально.

Клас `MainClass` містить наступні методи:

- `MainClass()` – конструктор, що відповідає за створення об'єктів;
- `main()` – головна програма, що відповідає за запуск застосунку.

Програмний код конструктора наступний:

```
public MainClass()
{
    mmc = new MyMidiClass(new About());
    intr = new Interface(mmc);
    mmc.setUpMidi();
}
```

Програмний код класу `MyNumberBeats` наступний:

```
package OS_NUZIP_MIDI;

public class MyNumberBeats {
    static final int COUNT_BEATS = 16;
}
```

Поле `COUNT_BEATS` даного класу встановлює розмір (кількість повторень інструмента) трека.

При розробці класу `Interface` було використано стандартні бібліотеки AWT для інтерфейсу користувача та Swing (частка бібліотеки базових класів – Java Foundation Classes).

Клас `Interface` містить наступні методи:

`Interface()` – конструктор, що відповідає за створення об'єктів;

`buildButtons()` – створення панелі кнопок;

`buildCheckBoxes()` – створення панелі прапорців, які відповідають за вибір послідовності;

`buildFrame()` – створення GUI рамки;

`buildGui()` – створення всієї GUI програми;

`buildInstrumentLabel()` – створення панелі з вказанням назви інструменту;

`displayFrame()` – встановлення розміру та положення вікна GUI;

`StartButtonListener()` – видаляє попередній трек, та починає програвати новий, базуючись на встановлених користувачем прапорцях;

`PauseButtonListener()` – призупинення;

`ResetButtonListener()` – призупинення та зняття виділення.

Код методу `buildButtons()`, що створює панель з трьома кнопками керування) наступний:

```
private void buildButtons() {
    JPanel buttonsPanel = new JPanel();
    buttonsPanel.setLayout(new
BoxLayout(buttonsPanel, BoxLayout.Y_AXIS));
    JButton startButton = new JButton("Start");
    JButton pauseButton = new JButton("Pause");
    JButton resetButton = new JButton("Reset");
```

```

        startButton.addActionListener (new
StartButtonListener());

        pauseButton.addActionListener (new
PauseButtonListener());

        resetButton.addActionListener (new
ResetButtonListener());

        buttonsPanel.add(startButton);
        buttonsPanel.add(pauseButton);
        buttonsPanel.add(resetButton);
        frame.add(BorderLayout.EAST, buttonsPanel);
    }

```

Класс MyMidiClass містить AWT методи:

TOOLS_NAMES – імена усіх інструментів;

TOOLS_KEYS – ключі (значення) усіх інструментів;

MyMidiClass() – конструктор, що відповідає за створення об'єктів;

buildTrackAndStart() – побудовує доріжку та починає

програвати її;

makeBeat() – програвання однієї ноти;

makeEvent() – програвання однієї події (початку або кінця звучання ноти);

makeTrack() – створення треку згідно обраних користувачем прапорців;

setUpMidi() – встановлення звукової системи midi;

startSequencer() – секвенсор починає програвання;

stopSequencer() – секвенсор зупинює програвання.

Програмний код методу buildTrackAndStart() наступний:

```

void buildTrackAndStart() {
    try {
        sequence.deleteTrack(track);
    }
}

```

```

        track = sequence.createTrack();
        makeTrack();
        sequencer.setSequence(sequence);

sequencer.setLoopCount(Sequencer.LOOP_CONTINUOUSLY);
    } catch (InvalidMidiDataException
invalidMidiDataException) {
        invalidMidiDataException.printStackTrace();
    }
}

```

Класс About містить наступні методи:

- About () – конструктор, що відповідає за створення об'єктів;
- addCheckBox () – призначає прапорець (чекбокс) у відповідне положення масиву;
- deselectCheckBox () – знімає прапорець, при умові що він вже стоїть;
- isCheckBoxSelected () – визначає чи встановлено прапорець користувачем.

Програмний код методу deselectCheckBox () наступний:

```

void deselectCheckBox(int a, int b) {
    if (isCheckBoxSelected(a, b)) {
        mycheckBoxArray[a][b].setSelected(false);
    }
}

```

Код програми розробленого застосунку наведений у Додатку А.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування програми

Розроблене програмне забезпечення призначене для програвання MIDI формату у заданій послідовності за допомогою набору інструментів.

Застосунок є кросплатформним та може працювати під керуванням ОС Windows, Mac та Linux. Для забезпечення роботи застосунку під керуванням ОС Windows, на комп'ютері має бути встановлене середовище розробки JDK (Java Development Kit).

4.2 Характеристики програми

Програма запускається за допомогою exe-файлу. Керування програмою відбувається з використанням графічного інтерфейсу. Завершити роботу програми можна за допомогою комбінації клавіш alt+f4 або через посередництво диспетчера задач.

Програму можна розповсюдити шляхом копіювання на usb-носії, використовуючи всесвітню мережу Інтернет, жорсткі диски та ін.

4.3 Вхідні та вихідні дані

Вхідними даними для розробленого застосунку є обрані користувачем команди (початок програвання, призупинення програвання, вибір та очищення послідовності та ін.).

Вихідними даними є програвання MIDI композиції.

4.4 Повідомлення програмісту

Повідомлень програмісту у програмі не передбачено.

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення програми

Розроблене програмне забезпечення призначене для відтворення MIDI композицій.

5.2 Умови використання програми

Розроблений застосунок є кросплатформним і може працювати під управлінням операційних систем Linux, Mac та сімейства Windows. Для роботи під керуванням ОС Windows, на комп'ютері має бути встановлене середовище розробки JDK.

Версія ОС обумовлює склад та системні вимоги до технічних засобів. У п. 1.3.4 наведена конфігурація, рекомендована для роботи з ОС Windows 10. Програмний застосунок потребує до 1 Мб вільного місця. Також він займає до 40 Мб оперативної пам'яті.

5.3 Виконання програми

Для запуску програмного застосунку необхідно запустити файл NUZP_BeatBox.exe. Після запуску програми оператору відкривається її головна форма, що наведена на рис. 5.1. Знаходячись на головній формі оператор може за допомогою чек боксів (прапорців) задати послідовність програвання бажаних інструментів (рис. 5.2)

За допомогою кнопок управління (рис. 5.2) оператор може формувати наступні команди:

- для того щоб почати програвання композиції треба натиснути кнопку «Start»;
- для зупинки програвання треба натиснути кнопку «Pause»;

- для очищення послідовності програвання треба натиснути кнопку «Reset».

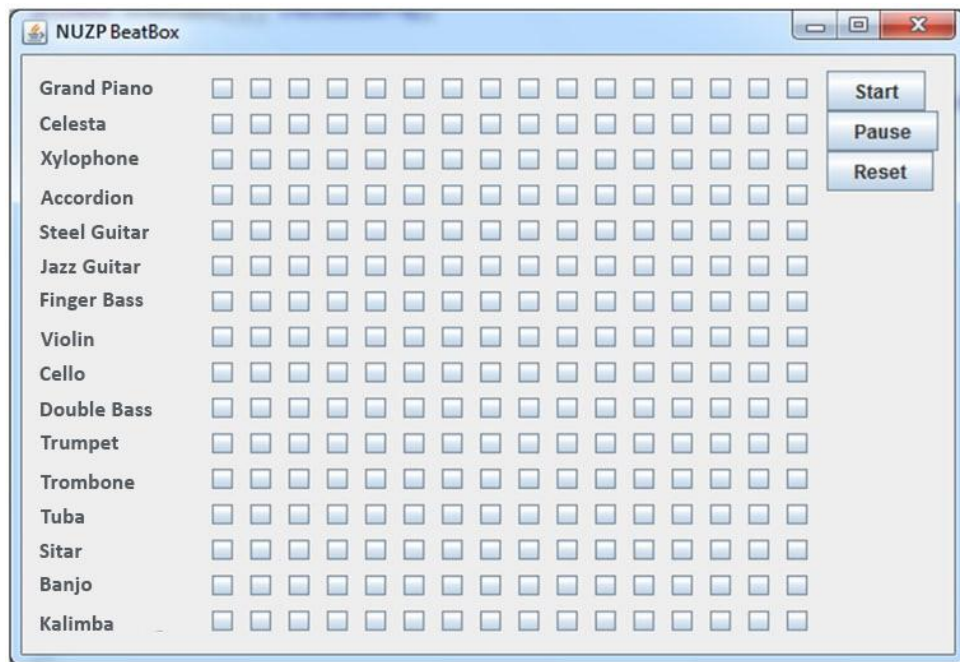


Рисунок 5.1 – Головне вікно програми



Рисунок 5.2 – Вибір послідовності програвання бажаних інструментів

Після очищення послідовності програвання, головне вікно програми буде мати вигляд наведений на рис. 5.1.

ВИСНОВКИ

За результатами дипломної кваліфікаційної роботи бакалавра було вирішено актуальну проблему розробки кросплатформного програвача створених користувачем MIDI композицій.

Для вирішення поставлених задач, в процесі виконання роботи було проведено огляд сфер застосування MIDI програвачів та принципів їх роботи. З'ясовано, що найбільш розповсюдженим способом синтезу звуків є субтрактивний синтез та розглянуто стандартний набір компонентів, необхідний для його реалізації.

Аналіз програмних аналогів показав, що найближчим аналогом програми, що розробляється є Acid Music Studio 11, яка дозволяє користувачу самостійно створювати мелодію із певного набору нот та певних музичних інструментів. Але цей застосунок є платною та не є кросплатформною, а працює лише під управлінням сімейства ОС Windows. Було прийнято рішення розробляти програму у вигляді десктопного застосунку, Для забезпечення вимоги ТЗ щодо кросплатформності, мовою розробки застосунку було обрано Java.

У відповідності з функціональними вимогами, зазначеними у ТЗ, розроблено структурну схему застосунку, а також блок-схему алгоритму роботи програми. Також було розглянуто основні класи програми та їх методи.

Програмна документація включає: керівництво програміста; керівництво оператора; опис програми та тексти основних програмних модулів.

Практичне використання розробленого програмного забезпечення дозволяє користувачу створювати та програвати музичні композиції, що представлені у MIDI форматі.

Галузь використання розробленого застосунку – музична індустрія.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Synthesizer [Electronic resource]. – Access mode: <https://en.wikipedia.org/wiki/Synthesizer>
2. Manning, P. Electronic and Computer Music / P. Manning. – Oxford: Oxford University Press, 1994. – 144 p.
3. Anderton, C. MIDI For Guitarists: A Crash Course In MIDI Effects Control. Sound on Sound / C. Anderton. – Access mode: http://www.soundonsound.com/sos/1995_articles/may95/midiforguitarists.html.
4. RCA Mark II Sound Synthesizer [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/RCA_Mark_II_Sound_Synthesizer.
5. Pinch, T., Trocco, F. Analog Days: The Invention and Impact of the Moog Synthesizer / T. Pinch, F. Trocco. – Harvard: Harvard University Press, 2004. – 104p.
6. Stevenson S. What Is the Time Signature of the Ominous Electronic Score of The Terminator? / S. Stevenson. – Access mode: <https://slate.com/culture/2014/02/the-time-signature-of-the-terminator-score-is-a-mystery-for-the-ages.html>.
7. Twells, J. The 14 most important synths in electronic music history – and the musicians who use them / J. Twells. – Access mode: <https://www.factmag.com/2016/09/15/14-most-important-synths/>.
8. Основы звукового синтеза [Электрон. ресурс]. – Режим доступа: <https://digitalmusicacademy.ru/lesson-synthesis-fundamentals>.
9. What is additive synthesis? The ultimate beginner's guide [Electronic resource]. – Access mode: <https://www.musicradar.com/news/what-is-additive-synthesis>.
10. Official MIDI Specifications [Electronic resource]. – Access mode: <https://www.midi.org/specifications>.
11. Standard MIDI-File Format Spec. 1.1, updated [Electronic resource]. – Access mode: <http://www.music.mcgill.ca/~ich/classes/mumt306/StandardMIDIfileformat.html>.

12. MIDI Tutorial [Electronic resource]. – Access mode: <https://www.cs.cmu.edu/~music/cmsip/readings/MIDI%20tutorial%20for%20programmers.html>.

13. Media Player Classic [Electronic resource]. – Access mode: <https://github.com/clsid2/mpc-hc>.

14. Media Player Classic – BE [Electronic resource]. – Access mode: <https://sourceforge.net/projects/mpcbe/>.

15. Acid Music Studio 11 [Electronic resource]. – Access mode: <https://www.magix.com/int/music/acid/acid-music-studio/>.

16. Sweet MIDI Player [Electronic resource]. – Access mode: <https://www.microsoft.com/uk-ua/p/sweet-midi-player/9nblggh4xvfp?activetab=pivot:overviewtab>.

17. Soundfont Midi Player 6.0 [Electronic resource]. – Access mode: <https://www.fileeagle.com/software/313/Soundfont-Midi-Player/>.

18. Framework Core MIDI [Electronic resource]. – Access mode: <https://developer.apple.com/documentation/coremidi/>.

19. Audio Toolbox [Electronic resource]. – Access mode: <https://ch.mathworks.com/products/audio.html>.

20. MIDI Device Interface [Electronic resource]. – Access mode: <https://ch.mathworks.com/help/audio/ug/midi-device-interface.html>.

21. Simple Java MIDI player followup [Electronic resource]. – Access mode: <https://codereview.stackexchange.com/questions/83509/simple-java-midi-player-followup>.

22. A Beginner's Guide To MIDI: What Is It? How Does It Work? | Musician's HQ [Electronic resource]. – Access mode: <https://musicianshq.com/a-beginners-guide-to-midi/>

ДОДАТОК А
Текст програми

A.1 Код файла MainClass.java

```
package OS_NUZIP_MIDI;

public class MainClass {

    Interface intr;
    MyMidiClass mmc;

    public MainClass()
    {
        mmc = new MyMidiClass(new About());
        intr = new Interface(mmc);
        mmc.setUpMidi();
    }

    public static void main(String[] args) {
        new MainClass();
    }

}
```

A.2 Код файла MyNumberBeats.java

```
package OS_NUZIP_MIDI;

public class MyNumberBeats {
    static final int COUNT_BEATS = 16;
}
```

A.3 Код файла Interface.java

```
package OS_NUZIP_MIDI;
//используем стандартные библиотеки GUI и подход из [21].
import javax.swing.WindowConstants;
import java.awt.Label;
import javax.swing.JButton;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import javax.swing.JPanel;
import javax.swing.BorderFactory;
import javax.swing.JCheckBox;
import javax.swing.JFrame;
import java.awt.BorderLayout;
import javax.swing.BoxLayout;
```

```

import java.awt.Dimension;

public class Interface {
    private MyMidiClass mmc;
    private About abt;
    private JFrame frame;

    public Interface(MyMidiClass mmc){
        this.mmc = mmc;
        abt = mmc.abt;
        buildGUI();
    }

    private void buildButtons(){
        JPanel buttonsPanel = new JPanel();
        buttonsPanel.setLayout(new BoxLayout(buttonsPanel,
        BoxLayout.Y_AXIS));
        JButton startButton = new JButton("Start");
        JButton pauseButton = new JButton("Pause");
        JButton resetButton = new JButton("Reset");
        startButton.addActionListener(new StartButtonListener());
        pauseButton.addActionListener(new PauseButtonListener());
        resetButton.addActionListener(new ResetButtonListener());

        buttonsPanel.add(startButton);
        buttonsPanel.add(pauseButton);
        buttonsPanel.add(resetButton);
        frame.add(BorderLayout.EAST, buttonsPanel);
    }

    private void buildCheckBoxes(){
        GridLayout grid = new
        GridLayout(MyMidiClass.TOOLS_NAMES.length,
        MyNumberBeats.COUNT_BEATS);
        grid.setVgap(0);
        grid.setHgap(2);
        JPanel checkBoxPanel = new JPanel(grid);
        for(int i = 0; i < MyMidiClass.mtls.length; i++){
            for(int j = 0; j < MyNumberBeats.COUNT_BEATS; j++) {
                JCheckBox checkBox = new JCheckBox();
                checkBoxPanel.add(checkBox);
                abt.addCheckBox(i, j, checkBox);
            }
        }
        frame.add(BorderLayout.CENTER, checkBoxPanel);
    }

    private void buildFrame(){
        frame = new JFrame("NUZP BeatBox");

        frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }
}

```

```

        frame.setMinimumSize(new Dimension(590, 365));
        JPanel panel = new JPanel(new BorderLayout());
        panel.setBorder(BorderFactory.createEmptyBorder(12, 12,
12, 12));
        frame.setContentPane(panel);
    }

    private void buildGUI() {
        buildFrame();
        buildInstrumentLabel();
        buildCheckBoxes();
        buildButtons();
        displayFrame();
    }

    private void buildInstrumentLabel() {
        JPanel namePanel = new JPanel();
        namePanel.setLayout(new BoxLayout(namePanel,
BoxLayout.Y_AXIS));
        for(int i = 0; i < MyMidiClass.mtls.length; i++){
            namePanel.add(new
Label(MyMidiClass.mtls[i].getName()));
        }
        frame.add(BorderLayout.WEST, namePanel);
    }

    private void displayFrame() {
        frame.pack();
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
    }

    private class StartButtonListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent ev){
            mmc.buildTrackAndStart();
            mmc.startSequencer();
        }
    }

    private class PauseButtonListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent ev){
            mmc.pauseSequencer();
        }
    }

    private class ResetButtonListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent ev){
            mmc.pauseSequencer();
        }
    }

```

```

        for(int i = 0; i < MyMidiClass.TOOLS_NAMES.length;
i++) {
            for(int j = 0; j < MyNumberBeats.COUNT_BEATS;
j++) {
                abt.deselectCheckBox(i, j);
            }
        }
    }
}
}

```

A.4 Код файла MyMidiClass.java

```

package OS_NUZP_MIDI;

import javax.sound.midi.Track;
import javax.sound.midi.Sequence;
import javax.sound.midi.InvalidMidiDataException;
import javax.sound.midi.ShortMessage;
import javax.sound.midi.MidiEvent;
import javax.sound.midi.MidiSystem;
import javax.sound.midi.MidiUnavailableException;
import javax.sound.midi.Sequencer;

public class MyMidiClass {
    private Sequencer sequencer;
    private Sequence sequence;
    private Track track;
    static final String[] TOOLS_NAMES = {"Grand Piano", "Celesta",
    "Xylophone", "Accordion", "Steel Guitar",
    "Jazz Guitar", "Finger Bass", "Violin", "Cello",
    "Double Bass", "Trumpet", "Trombone", "Tuba", "Sitar",
    "Banjo", "Kalimba"};
    private static final int[] TOOLS_KEYS = {3, 9, 14, 22, 26, 27,
34, 41, 43, 44, 57, 58, 59, 105, 106, 109};
    static final MyTools[] mtls = new MyTools[TOOLS_NAMES.length];
    About abt;

    public MyMidiClass(About abt) {
        this.abt = abt;
        setUpInstruments();
    }
    //используем стандартный подход для проигрывания из [21].
    void buildTrackAndStart() {
        try {
            sequence.deleteTrack(track);
            track = sequence.createTrack();
            makeTrack();
            sequencer.setSequence(sequence);
            sequencer.setLoopCount(Sequencer.LOOP_CONTINUOUSLY);

```

```

        } catch (InvalidMidiDataException invalidMidiDataException) {
            invalidMidiDataException.printStackTrace();
        }
    }

    private void makeBeat(int key, int tick) throws
InvalidMidiDataException{
        track.add(makeEvent(ShortMessage.NOTE_ON, 9, key, 100,
tick));
        track.add(makeEvent(ShortMessage.NOTE_OFF, 9, key, 100,
tick + 2));
    }

    private MidiEvent makeEvent(int command, int channel, int one,
int two, int tick) throws InvalidMidiDataException{
        ShortMessage a = new ShortMessage();
        a.setMessage(command, channel, one, two);
        return new MidiEvent(a, tick);
    }

    private void makeTrack() throws InvalidMidiDataException{
        for(int i = 0; i < mtls.length; i++){
            for(int j = 0; j < MyNumberBeats.COUNT_BEATS; j++){
                if(abt.isCheckBoxSelected(i, j)){
                    makeBeat(mtls[i].getKey(), j);
                }
            }
        }

        track.add(makeEvent(ShortMessage.PROGRAM_CHANGE, 9, 1, 0,
MyNumberBeats.COUNT_BEATS - 1));
    }

    static void setUpInstruments(){
        for(int i = 0; i < mtls.length; i++){
            mtls[i] = new MyTools(TOOLS_NAMES[i], TOOLS_KEYS[i]);
        }
    }

    void setUpMidi(){
        try {
            sequencer = MidiSystem.getSequencer();
            sequencer.open();
            sequence = new Sequence(Sequence.PPQ, 4);
            track = sequence.createTrack();
        } catch (InvalidMidiDataException invalidMidiDataException) {
            invalidMidiDataException.printStackTrace();
        } catch (MidiUnavailableException midiUnavailableException) {
            //когда требуемый компонент MIDI не может быть открыт или создан,
            потому что это недоступно.
            midiUnavailableException.printStackTrace();
        }
    }
}

```

```

void startSequencer() {
    sequencer.start();
}

void pauseSequencer() {
    sequencer.stop();
}

```

A.5 Код файла About.java

```

package OS_NUZP_MIDI;

import javax.swing.JCheckBox;

public class About {
    private JCheckBox[][] mycheckBoxArray;

    public About() {
        mycheckBoxArray = new
        JCheckBox[MyMidiClass.TOOLS_NAMES.length][MyNumberBeats.COUNT_BEAT
        S];
    }

    void addCheckBox(int x, int y, JCheckBox checkBox) {
        mycheckBoxArray[x][y] = checkBox;
    }

    void deselectCheckBox(int x, int y) {
        if(isCheckBoxSelected(x, y)) {
            mycheckBoxArray[x][y].setSelected(false);
        }
    }

    boolean isCheckBoxSelected(int x, int y) {
        return mycheckBoxArray[x][y].isSelected();
    }
}

```

ДОДАТОК Б
Слайди презентації

ДИПЛОМНА РОБОТА

РОЗРОБКА ПРОГРАМИ MIDI ПРОГРАВАЧА

Розробив: ст. групи КНТ-218
Керівник: доцент, к.т.н.

Омельченко Є.В.
Сердюк С.М.

Рисунок Б.1 – Слайд №1

Мета роботи

розробка кросплатформної програми, що дозволяє створювати та програвати музичні композиції, що представлені у MIDI форматі

Об'єкт дослідження – процеси синтезу звуку

Предмет дослідження – методи побудови та відтворення MIDI композицій



НУ "ЗП", Кафедра програмних засобів

Рисунок Б.2 – Слайд №2

Задачі, що вирішуються

- огляд сфер застосування MIDI програвачів та принципів їх роботи;
- аналіз програмних аналогів;
- розробка технічного завдання;
- визначення компонентів системи;
- розробка алгоритму роботи програми;
- програмна реалізація застосунку;
- розробка необхідної документації.



НУ "ЗП", Кафедра програмних засобів

3

Рисунок Б.3 – Слайд №3

Аналіз програмних аналогів

Параметр	Windows Media Player	Acid Music Studio 11	Sweet Midi Player	Soundfont Midi Player 6.0
1. Можливість створення MIDI файлів з набору нот	-	+	-	+
2. Можливість вибору музичного інструменту для відтворення MIDI мелодії	-	+	-	-
3. Можливість застосування ефектів	-	+	+	+
4. Кросплатформність	-	-		-
5. Ціна	Безкоштовна	59,99 €	\$29,95	Безкоштовна



НУ "ЗП", Кафедра програмних засобів

4

Рисунок Б.4 – Слайд №4

MIDI повідомлення

Структура повідомлення **NOTE ON**:

- байт стану: 1001 SSSS;
- байт даних 1: 0TTT TTTT;
- байт даних 2: 0VVV VVVV.

Структура повідомлення **NOTE OFF**:

- байт стану: 1000 SSSS;
- байт даних 1: 0TTT TTTT;
- байт даних 2: 0VVV VVVV.



НУ "ЗП", Кафедра програмних засобів

Рисунок Б.5 – Слайд №5

5

Приклад числових кодів деяких музичних інструментів

Група	Номер	Оригінальна назва	Переклад
1	2	3	4
Piano (Фортепіано)	1	Acoustic piano	акустичний рояль
	3	Grand piano	великий рояль
	7	Harpsichord	клавесин
Chromatic Percussion (Хроматичні ударні)	9	Celesta	челеста
	10	Glockenspiel	дзвіночки
	14	Xylophone	ксилофон
Organ (Орган)	17	Hammond	орган Хаммонда
	22	Accordion	акордеон
	23	Harmonica	гармоніка
Guitar (Гітара)	25	Nylon Guitar	акустична гітара (з нейлоновими струнами)

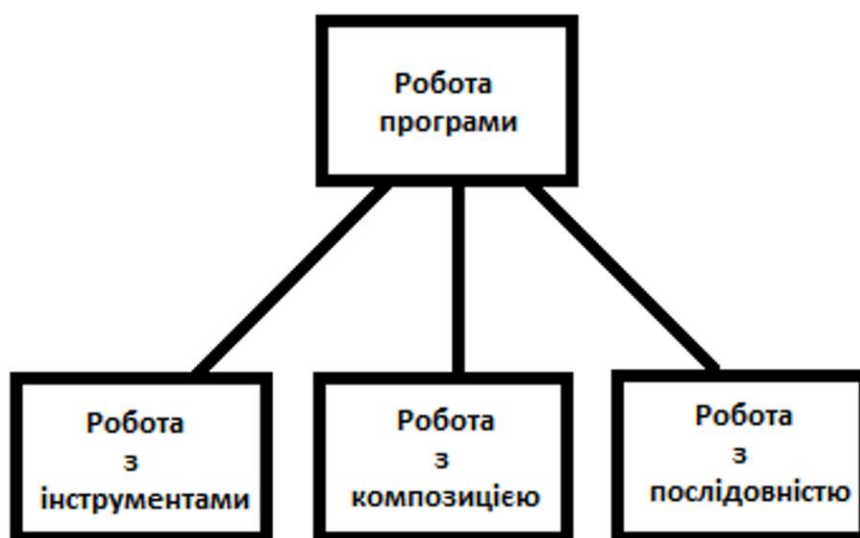


НУ "ЗП", Кафедра програмних засобів

Рисунок Б.6 – Слайд №6

6

Функціональна структура програми

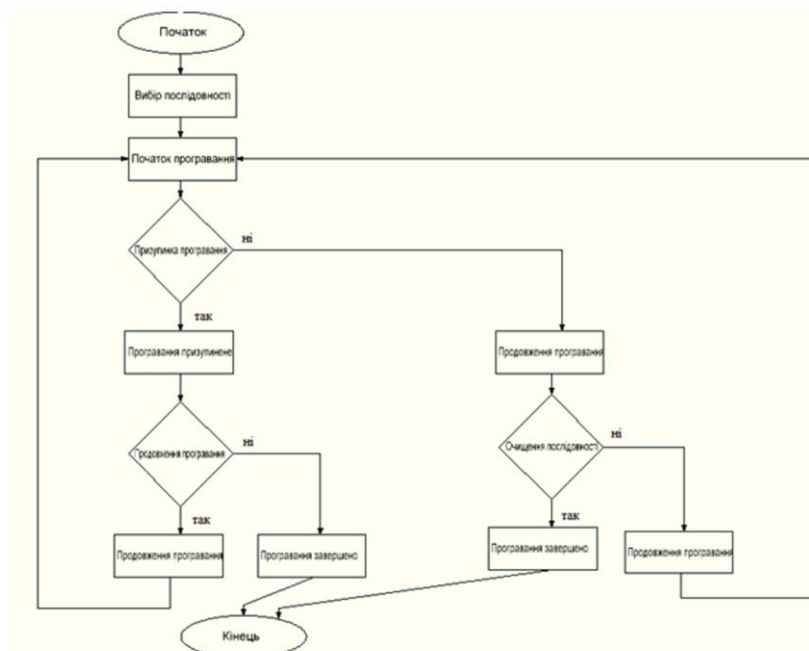


НУ "ЗП", Кафедра програмних засобів

7

Рисунок Б.7 – Слайд №7

Блок-схема алгоритму роботи програми



НУ "ЗП", Кафедра програмних засобів

8

Рисунок Б.8 – Слайд №8

Опис класу MyMidiClass

Назва методу, змінної	Призначення
TOOLS_NAMES	імена усіх інструментів
TOOLS_KEYS	ключі (значення) усіх інструментів
MyMidiClass (	конструктор, що відповідає за створення об'єктів
buildTrackAndStart ()	побудовує доріжку та починає програвати її
makeBeat ()	програвання однієї ноти
makeEvent ()	програвання однієї події (початку або кінця звучання ноти)
makeTrack ()	створення треку згідно обраних користувачем прапорців
setUpMidi ()	встановлення звукової системи midi
startSequencer ()	секвенсор починає програвання
stopSequencer ()	секвенсор зупинює програвання

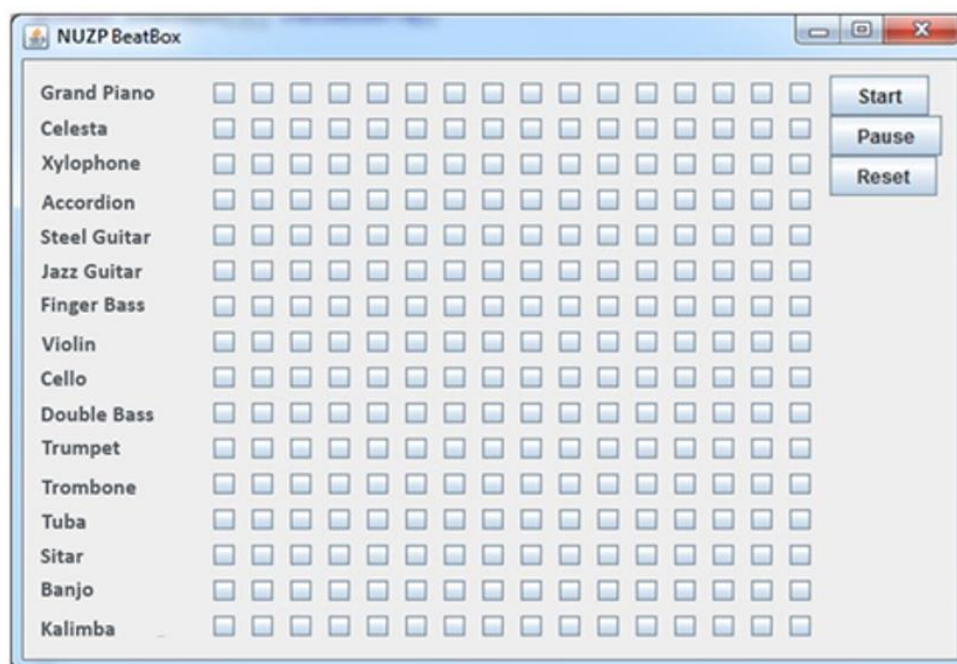


НУ “ЗП”, Кафедра програмних засобів

Рисунок Б.9 – Слайд №9

9

Головне вікно програми



НУ “ЗП”, Кафедра програмних засобів

Рисунок Б.10 – Слайд №10

10

Вибір послідовності програвання інструментів



НУ "ЗП", Кафедра програмних засобів

11

Рисунок Б.11 – Слайд №11

Висновки

- проведено огляд сфер застосування MIDI програвачів та принципів їх роботи;
- проведено аналіз програмних аналогів;
- розроблено технічне завдання;
- визначено компоненти системи;
- розроблено алгоритми роботи програми;
- програмно реалізовано застосунок;
- розроблено необхідну документацію.



НУ "ЗП", Кафедра програмних засобів

12

Рисунок Б.12 – Слайд №12

**Доклад закінчено,
дякую за увагу!**



НУ "ЗП", Кафедра програмних засобів

Рисунок Б.13 – Слайд №13

13