Procesory sygnałowe
Etap 3 - Dokumentacja aplikacji

Mikołaj Saramonowicz 252964
Dawid Szymczyna 252920
Grupa 8.15 Poniedziałek TN

21 grudnia 2022

1 Ogólny opis wykonanego programu

Napisany pod systemem Windows program procesora dźwiękowego pozwala na wgrywanie i odtwarzanie dowolnego pliku z rozszerzeniem .WAV, jak i jego zapisanie pod wybraną przez użytkownika nazwą. Na tak wgranym pliku użytkownik może wykonać 6 różnych filtrów oraz po zaznaczeniu odpowiedniego pola, ich dowolne kombinacje. Jeżeli odpowiednie pola nie zostaną zapełnione w programie, wykonane zostaną operacje na przykładowym pliku, który zawarty jest w ZIPie razem z programem ("test-audio.wav").

Program dodatkowo dba o to, żeby wpisane przez użytkownika parametry znajdowały się w odpowiednim zakresie i w przypadku wykrycia nieprawidłowości informuje użytkownika lub sam zadaje odpowiednią wartość.

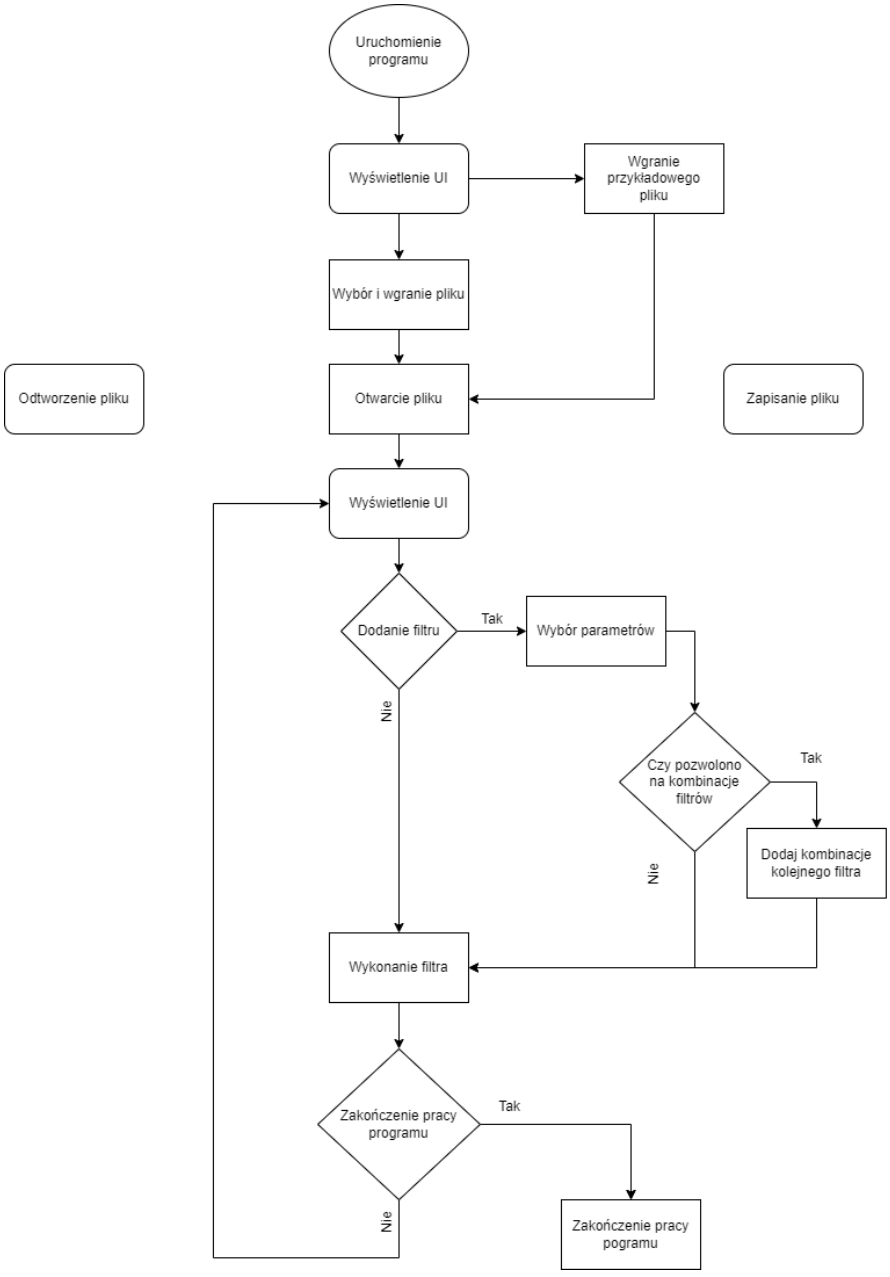
Wymienione funkcjonalności zrealizowano za pomocą języka C++. W projekcie wykorzystano wybrane gotowe biblioteki i narzędzia, ułatwiające operowanie na dźwiękach oraz sterowanie logiką aplikacji.

Bardziej szczegółowe opisy działania zastosowanych mechanizmów można znaleźć w załączonym do sprawozdania kodzie źródłowym projektu.

2 Wykorzystane rozwiązania

- **Program Audacity** - otwarte wieloplatformowe narzędzie do obróbki dźwięku.
  W projekcie wykorzystano jego funkcjonalności oraz kod źródłowy do sprawdzenia poprawności działania efektów.
- **Framework QT** - popularny, aktywnie rozwijany i powszechnie dostępny zestaw narzędzi deweloperskich umożliwiający sprawne generowanie przejrzystych interfejsów graficznych dla aplikacji w języku C++ i Python. W procesorze dźwiękowym posłuży do wygenerowania przycisków wywołujących wybrane operacje po kliknięciu kursorem.
- **Biblioteka Audiofile** - rozbudowane narzędzie open-source zawierające zestaw funkcji umożliwiających wczytywanie dźwięków w formacie .WAV i zapisywanie ich w postaci znormalizowanej tablicy, znacznie ułatwiającej wykonywanie operacji matematycznych. Biblioteka pozwala także na zapis nowego dźwięku i ręczną edycję parametrów pliku.
- **Windowsowe narzędzie Playsound** - dostarczona przez OS funkcja z biblioteki Windows.h pozwalająca na zagranie dźwięku ze wskazanego pliku w obecnym katalogu. Rozwiązanie zastosowano z powodu ciągle niezrealizowanego wsparcia dla stabilnej wersji komend obsługi dźwięków w QT 6.4.1.

3 Schemat działania programu



Rysunek 1: Schemat architektury wysokopoziomowej

4 Opis wykonanych efektów/filtrów

- Amplify — zwiększenie lub zmniejszenie głośności wybranego dźwięku. Użytkownik będzie miał możliwość stopniowania operacji za pomocą suwaka zwiększającego lub zmniejszającego współczynnik wzmocnienia w skali logarytmicznej (dB). Zastosowanie efektu wymaga dodatkowej weryfikacji maksymalnego wzmocnienia niepowodującego zaburzenia wzajemnego stosunku między amplitudami, co zostało zrealizowane w kodzie poprzez odpowiednie porównania.

- Reverse — przetworzony dźwięk będzie słyszany od tyłu. Brak konieczności konfigurowalnych parametrów. Najprostszy z zaproponowanych efektów, oparty na podstawowej operacji odwracania kolejności próbek.

- Echo — efekt powoduje kilkukrotne powtórzenie dźwięku co określony czas i z każdorazowo zmienioną głośnością. Zakłada się, że użytkownik będzie mógł wybrać współczynnik wygaszania/wzmacniania głośności oraz przerwę czasową pomiędzy kolejnymi powtórzeniami. Realizację fizyczną oparto na połączeniu operacji dodawania wartości do próbki z wykładniczo zmieniającym się mnożnikiem.

- Fade out — zmniejsza głośność wybranego dźwięku poprzez ustalony okres, który pozostał do jego końca — efekt stopniowego wygaszania (podany okres musi być krótszy niż sam dźwięk). Fizyczną realizację oparto na podzieleniu 1 (brak wzmocnienia) przez liczbę próbek przypadających na czas trwania efektu, w ten sposób uzyskując współczynnik zmiany wzmocnienia na próbkę, pozwalający zastosować liniowe przejście do wartości 0.

- Fade in — zwiększa głośność wybranego dźwięku poprzez ustalony okres, który pozostał do jego końca (podany okres musi być krótszy niż sam dźwięk). Efekt zastosowano zamiast początkowo planowanego filtru dolnoprzepustowego z powodu trudności na etapie implementacji oraz wyraźnej analogii implementacji do efektu "Fade out".

- Change speed — zwiększa lub zmniejsza szybkość i czas trwania dźwięku, wykorzystując wskazany przez użytkownika mnożnik, pośrednio wpływa również na występujące częstotliwości. Efekt oparto na matematycznej operacji tymczasowej zmiany częstotliwości próbkowania dźwięku.

5 Struktura projektu

W celu zaoszczędzenia na czasie zastosowano bardzo uproszczony podział projektu na poszczególne pliki. W ramach potencjalnego dalszego rozwoju projektu zostanie to usprawnione, aby zwiększyć czytelność kodu.

- **Audiofile.h** - zawiera definicje funkcji umożliwiających wczytanie dźwięku do znormalizowanej tablicy, określenie jego podstawowych parametrów oraz zapisanie go ponownie.

- **MainWindow.h** - zawiera deklaracje funkcji związanych z interfejsem graficznym użytkownika.

- **Main.cpp** - zawiera polecenie otwarcia i zamknięcia aplikacji okienkowej.

- **MainWindow.cpp** - zawiera definicje funkcji związanych z interfejsem graficznym użytkownika, w tym wszystkie efekty i udostępnione polecenia ("rdzeń" całego projektu).

Pozostałe pliki zawarte w archiwum z projektem zostały wygenerowane przez narzędzie QT i nie zawierają kodu źródłowego.

6 Funkcje

Punkt zawiera wybrane, najważniejsze wykonane przez nas funkcje, wraz z krótkim opisem ich działania i wpisywanymi do nich parametrami. W spisie nie zawarto funkcji generowanych automatycznie przez framework do obsługi przycisków.

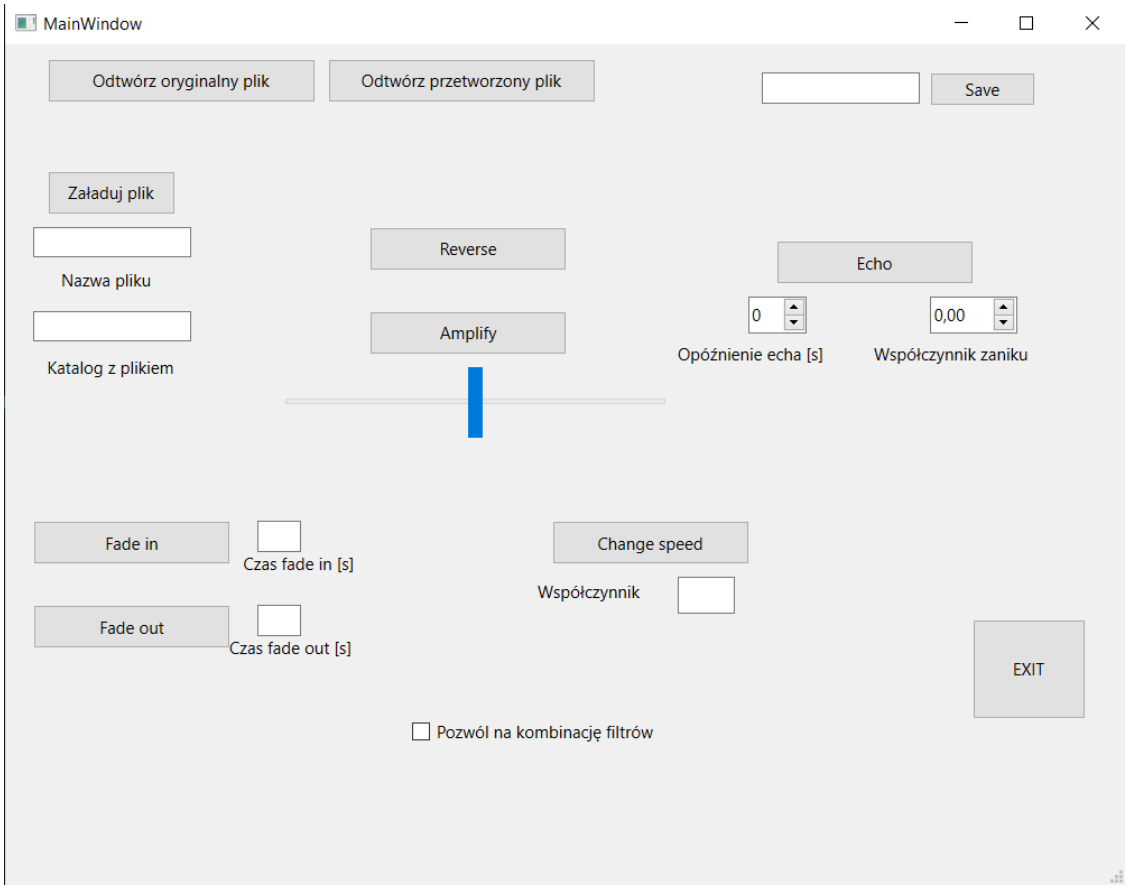
- **void loadAudioFileAndPrintSummary()** - wczytuje wybrany wcześniej przez użytkownika dźwięk i wypisuje w konsoli jego podstawowe parametry, takie jak np. długość trwania. Może zwrócić błąd w przypadku uszkodzenia pliku.
- **void loadAudioFileAndSaveIt(std::string fileNameSave)** - pobiera wpisaną przez użytkownika nazwę pliku, a następnie tworzy plik o takiej nazwie i zapisuje w nim aktualnie wczytany dzwięk.
- **void loadChoosenAudioFileAndPrintSummary(std::string directory, std::string fileName)** - pobiera wpisaną przez użytkownika nazwę pliku (z rozszerzeniem .WAV), wraz ze ścieżką do katalogu, w którym ten się znajduje, a następnie kopiuje ten plik do aktualnie używanego katalogu
- **void loadAudioFileAndAplify()** - wykonuje efekt 'Amplify' na załadowanym dźwięku biorąc pod uwagę położenie suwaka na interfejsie graficznym. Może zwrócić błąd w przypadku uszkodzenia pliku.
- **void loadAudioFileAndFadeIn(float fadetime)** - wykonuje efekt 'Fade in' na załadowanym dźwięku biorąc pod uwagę podany przez użytkownika czas trwania (w sekundach). Może zwrócić błąd w przypadku uszkodzenia pliku lub niepoprawnego argumentu wejściowego.
- **void loadAudioFileAndFadeOut(float fadetime)** - - wykonuje efekt 'Fade out' na załadowanym dźwięku biorąc pod uwagę podany przez użytkownika czas trwania (w sekundach). Może zwrócić błąd w przypadku uszkodzenia pliku lub niepoprawnego argumentu wejściowego.
- **void loadAudioFileAndReverse()** - wykonuje efekt 'Reverse' na załadowanym dźwięku. Może zwrócić błąd w przypadku uszkodzenia pliku.
- **void changeSpeedAudioFile(float frequency)** - wykonuje efekt 'Change speed' na załadowanym dźwięku biorąc pod uwagę podany przez użytkownika współczynnik. Może zwrócić błąd w przypadku uszkodzenia pliku lub niepoprawnego argumentu wejściowego.
- **void add_echoes()** - wykonuje efekt 'Echo' na załadowanym dźwięku biorąc pod uwagę podane przez użytkownika opóźnienie echa (w sekundach) i współczynnik zaniku. Może zwrócić błąd w przypadku uszkodzenia pliku lub niepoprawności któregokolwiek z argumentów wejściowych.
- **std::string MainWindow::on_Button_directory_clicked()** - po wpisaniu w pole podpisane "Nazwa pliku" oraz "Katalog z plikiem" odpowiednich danych, wczytuje nazwę oraz ścieżkę od pliku.
- **void MainWindow::on_checkBox_toggled(bool checked)** - pozwala na wykonywanie sekwencji filtrów.

7 Klasy

- **class MainWindow : public QMainWindow** - klasa posiadające funkcje odpowiedzialne za obsługę interfejsu użytkownika. Są to zarówno obsługa samych przycisków i suwaków, które użytkownik musi w celu wykonania danego działania nacisnąć, jak i pól tekstowych, w których użytkownik musi podać liczbę z danego zakresu lub nazwę.
- **class AudioFile** - klasa posiadające funkcje odpowiedzialne za obsługę wejściowego pliku z rozszerzeniem .WAV. Posiada również kilka funkcji, które wywołane pozwalają zwrócić informację o podstawowych parametrach wczytanego pliku (takich jak np. liczba kanałów czy długość dźwięku) oraz je modyfikować (np. zmiana długości dźwięku, tj. przycięcie lub wypełnienie ciszą).

8 Instrukcja użytkowania

Po uruchomieniu aplikacji wyświetli się okno zawierające przyciski pozwalające na skorzystanie z jej funkcjonalność. Dla uproszczenia implementacji wszystkie możliwe opcje wyświetlono na jednym panelu.



Rysunek 2: Interfejs użytkownika

Aby poprawnie skorzystać z programu, należy zastosować się do poniższych zasad:

- W pierwszej kolejności niezbędne jest podanie nazwy pliku i jego katalogu oraz naciśnięcie przycisku "Załaduj plik"
- Odtworzenie załadowanego na początku pliku lub wersji z nałożonym efektem umożliwiają przyciski w lewym górnym rogu okna

- Domyślnie każdy efekt wykonywany jest na pliku oryginalnym. Aby pozwolić na nałożenie kilku efektów naraz należy po nałożeniu pierwszego efektu zaznaczyć opcję "Pozwól na kombinację filtrów".
- Każdy z przycisków z opisanym efektem pozwala na nałożenie go na dźwięk. Przed wyborem należy pamiętać o podaniu jego parametrów w celu uniknięcia błędów programu. Przyciski w lewym górnym rogu ekranu umożliwiają sprawne porównanie oryginalnego dźwięku z edytowanym.
- Aby zapisać uzyskany efekt, należy podać ścieżkę nowego pliku i nacisnąć przycisk "Save"
- Wyjście z aplikacji umożliwia "X" lub przycisk "EXIT"

8.1 Uwagi dotyczące poszczególnych efektów

- Polecenia "Fade out" i "Fade in" w przypadku podania niepoprawnego czasu trwania wykonają się dla jego minimalnej wartości (10% długości dźwięku) lub maksymalnej (100% długości dźwięku).
- Polecenie "Amplify" jest regulowane za pomocą suwaka oraz posiada wbudowaną ochronę przed przesadnym wzmocnieniem, tj. takim, które uniemożliwia zachowanie stałego wzajemnego stosunku próbek. W programie Audacity można tę opcję wyłączyć, co skutkuje powstaniem zniekształconego dźwięku.
- Polecenie "Echo" wymaga, aby opóźnienie echa było dodatnie oraz krótsze niż długość samego dźwięku. Ponadto należy bardzo uważać przy wyborze dużych wartości współczynnika zaniku (1.3 i wyżej). Przy wielokrotnym powtórzeniu dźwięk zostanie wzmocniony wykładniczo, co z racji braku zabezpieczeń jak przy funkcji "Amplify" może powodować znaczne zniekształcenia. W bieżącej wersji polecenie nie wpływa na długość samego dźwięku i wymaga ręcznego dodania ciszy po jego zakończeniu (tak jak w programie Audacity).
- Polecenie "Change speed" wymaga, aby współczynnik przyspieszenia/spowolnienia był liczbą dodatnią. Nie posiada żadnych innych ograniczeń — to jakie wartości są fizycznie do zrealizowania, zależy od ilości próbek w danych dźwięku.

9 Podsumowanie

Napisany program pomimo prostego interfejsu użytkownika oraz kilku nienaprawionych błędów technicznych i stylistycznych spełnia swoją funkcję i stanowi dobrą podstawę do napisania w pełni funkcjonalnego procesora dźwięku będącego alternatywą dla programu Audacity.

10 Bibliografia

- `https://manual.audacityteam.org/` [dostęp 3.12.22]
- `https://github.com/audacity/audacity` [dostęp 3.12.22]
- `https://github.com/adamstark/AudioFile` [dostęp 3.12.22]
- `https://www.qt.io/` [dostęp 3.12.22]