Informatyka, studia dzienne, mgr II st. semestr I

Przetwarzanie obrazu i dźwięku 2020/2021

Prowadzący: dr inż. Bartłomiej Stasiak poniedziałek, 12:15

Data oddania: _____ Ocena: _____

Aleksander Dominiak 234049

Dominik Andrzejczak 234038

Zadanie 4:

Filtracja sygnału dźwiękowego w dziedzinie częstotliwości.

1. Cel

Celem niniejszego zadania było zaprojektowanie aplikacji pozwalającej na wczytanie dźwięku z pliku *.wav oraz umożliwienie filtracji sygnału dźwiękowego w dziedzinie częstotliwości oraz dziedzinie czasu z zastosowaniem filtra o skończonej odpowiedzi impulsowej.

2. Wprowadzenie

Wykonanie zadania wiązało się z implementacją operacji wczytania dźwięku, które oparte zostało na bibliotece nAudio dostępnej w pakiecie nuget w środowisku Visual Studio. Do wykonania części filtrowania w dziedzinie częstotliwości potrzebna była implementacja DFT. Zaimplementowano jej szybszą wersję FFT:

FFT

Szybkie przekształcenie Fouriera (FFT) to bardzo wydajny algorytm implementujący dyskretne przekształcenie Fouriera (DFT).

DFT to transformata Fouriera wyznaczona dla sygnału próbkowanego, a więc dyskretnego. Przekształca ona skończony ciąg próbek sygnału ($a_0, a_1, ..., a_{N-1}$), $a_i \in \mathrm{R}$ w ciąg harmoniczny ($A_0, A_1, ..., A_{N-1}$), $A_i \in \mathrm{C}$, zgodnie ze wzorem:

$$A_k = \sum_{n=0}^{N-1} a_n w_N^{-kn},\ 0 \leqslant k \leqslant N-1,$$

$$w_N = e^{i\frac{2\pi}{N}},$$

gdzie:
i – jednostka urojona,
k – numer harmoniczny,
n – numer próbki sygnału,
a_n – wartość próbki sygnału,

N – liczba próbek
Przekształcenie odwrotne wyraża się wzorem:

$$a_n = \frac{1}{N} \sum_{k=0}^{N-1} A_k w_N^{kn},\ 0 \leqslant n \leqslant N-1.$$

FFT pozwala na szybsze obliczenie DFT, najpopularniejszym wariantem FFT jest ten o podstawie 2, wymagający, by rozmiar DFT był całkowitą potęgą liczby dwa.

DFT do obliczenia wymaga wykonania N2 mnożeń. Dodatkowym problemem jest fakt, że są to mnożenia liczb zespolonych, znacznie zużywające moc obliczeniową procesora. W przypadku FFT liczba mnożeń spada do około N/2 * $\log_2$N, przy czym część z nich wymaga jedynie zmiany znaku.

Algorytm FFT korzysta z podanych heurystyk budowania szybkich algorytmów:

- „dziel i rządź” – podział zadania na mniejsze fragmenty,
- rekurencja,
- programowanie dynamiczne.

FILTR O SKOŃCZONEJ ODPOWIEDZI IMPULSOWEJ

Do zdefiniowania filtru posłużono się metodą okna, która określa kolejne współczynniki filtru w dziedzinie czasu.

$$h(k) = \begin{cases} \frac{2f_c}{f_s} & \text{, dla } k = \frac{L-1}{2} \\ \frac{\sin\left(\frac{2\pi f_c}{f_s}\left(k-\frac{L-1}{2}\right)\right)}{\pi\left(k-\frac{L-1}{2}\right)} & \text{, w przeciwnym wypadku} \end{cases}$$

gdzie $k = 0, 1, ..., L-1$.

Gdzie:

L – rozmiar filtra
fc – częstotliwość odcięcia
fs – częstotliwość próbkowania

Otrzymano L-1 kolejnych współczynników odpowiedzi impulsowej, które następnie wymnożono przez okno (prostokątne, von Hanna lub Hamminga) aby poprawić (wygładzić) charakterystykę filtru.

Najprostsze okno to okno prostokątne zdefiniowane za pomocą poniższego wzoru:

$$w(n) = \begin{cases} 1 & \text{, dla } n = 0, 1, ..., M-1\ , \\ 0 & \text{, w przeciwnym wypadku} \end{cases}$$

Kolejne okna von Hanna i Hamminga wyrażają się wzorami:

a) von Hann

$$w_{Hann}(n) = \frac{1}{2} - \frac{1}{2}\cos\left(\frac{2\pi n}{M-1}\right)$$

b) Hamming

$$w_{Hamming}(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{M-1}\right)$$

Dodatkowo okna von Hanna i Hamminga możemy otrzymać nie prawidłowe wartości na początku i końca okna. Aby je poprawić zmniejszamy ich wartości o połowę.

Przed przystąpieniem do filtracji wczytany sygnał dźwiękowy dzielimy na równej długości ramki określone przez użytkownika. Zdefiniowano również możliwość przesunięcia ramek (hop size) tak aby otrzymać ciągły sygnał wyjściowy.

DZIEDZINA CZASU

W dziedzinie czasu sygnał wejściowy na początku i na końcu uzupełniamy zerami:

$$\begin{array}{lllllll} x(-1) & = & x(-2) & = & \ldots & = & x(-(L-1)) & = & 0 \\ x(K) & = & x(K+1) & = & \ldots & = & x(K+L-2) & = & 0 \end{array}$$

Wynik operacji filtracji w dziedzinie czasu to zwykły splot sygnału wejściowego i współczynników odpowiedzi impulsowej:

$$y(n) = \sum_{k=0}^{L-1} x(n-k)h(k)$$

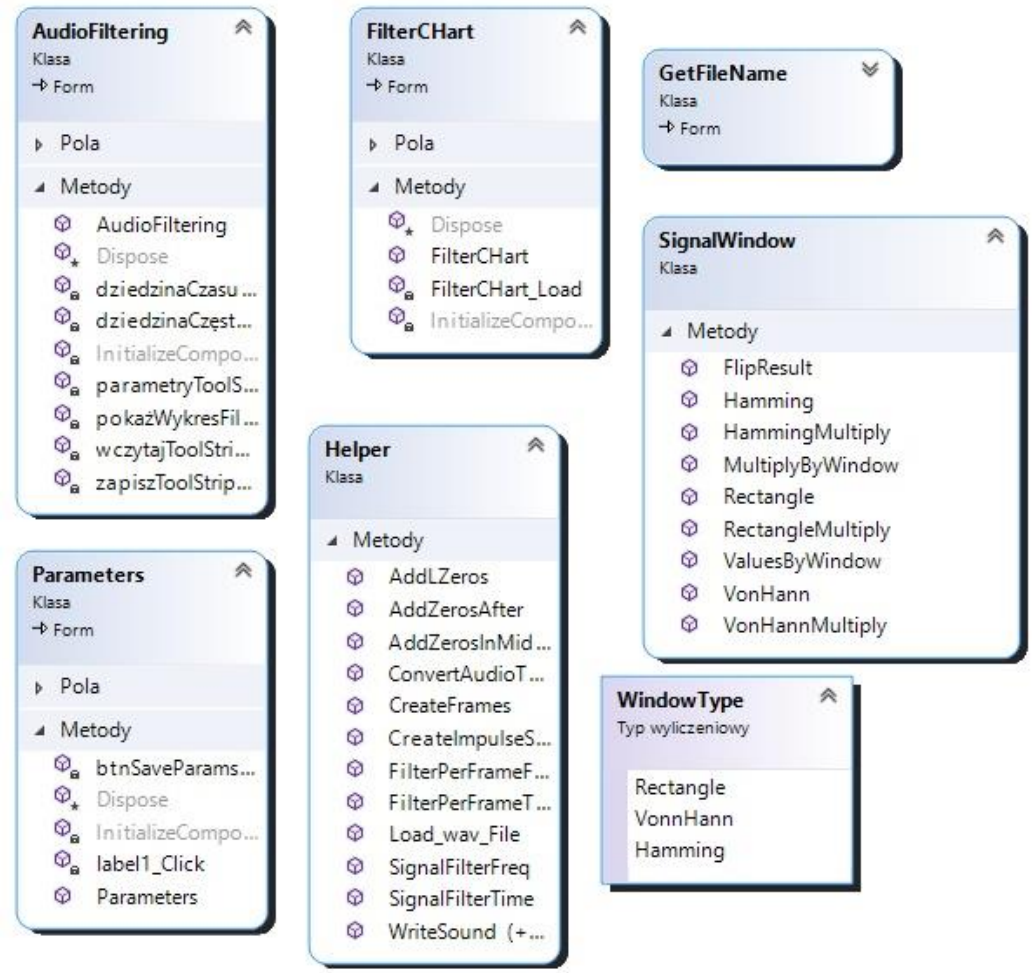
DZIEDZINA CZĘSTOTLIWOŚCI

W dziedzinie częstotliwości również uzupełniamy sygnał zerami. Różnica polega na dodaniu zer na końcu sygnału. W tym przypadku musimy również dodać zera w tablicy współczynników odpowiedzi impulsowej. Jeżeli filtr ma być przyczynowy to dodajemy zera na końcu, w przeciwnym wypadku w środku filtru.
Uzupełniania zerami zapewniają równą długość obu widm, które otrzymujemy za pomocą operacji FFT zarówno dla sygnału wejściowego jak i filtru.
Wynikiem filtracji jest iloczyn obu widm pamiętając o mnożeniu zarówno części rzeczywistej jak i urojonej.
Ostatecznym wynikiem dla każdej ramki jest odwrotna transformata Fouriera IFFT.

Wszystkie operacje wykonujemy niezależnie dla każdego okna sygnału. Ostateczny wynik otrzymujemy sumując wyniki dla kolejnych próbek.

3. Opis implementacji

Aplikacja została napisana w języku C# z użyciem frameworka Windows Forms.



AudioFiltering to główne okno programu służące do interakcji użytkownika z zaimplementowanymi funkcjonalnościami

FilterCHart – okno pozwalające wyświetlić wykres współczynników filtru.

GetFileName – okno pozwalające podać nazwę pliku dla zapisywanego dźwięku.

Parameters – okno edycji parametrów (rozmiar okna sygnału, rozmiar filtru, częstotliwość odcięcia, przesunięcie ramki, rodzaj okna)

SignalWindow – zaimplementowane odpowiednie wzory okien

Helper – główna klasy dostępnych operacji. SignalFilterTime – filtracja w dziedzinie czasu, SignalFilterFreq – filtracja w dziedzinie częstotliwości, CreateImpulseSignal – pobranie kolejnych wartości odpowiedzi impulsowej.

4. Materiały i metody

Do weryfikacji poprawności działania aplikacji wykorzystano następujące pliki audio:

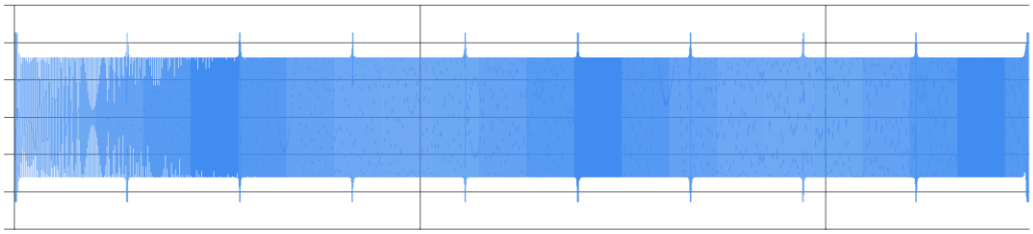
chirp_100Hz_1000Hz_lin.wav
chirp_100Hz_1000Hz_log.wav

Każdy sygnał filtrowano dla dwóch zestawów parametrów:

Rozmiar ramki:
a) 512
b) 1024
Przesunięcie ramki:
a) 256
b) 623
Rozmiar filtru:
a) 511
b) 911
Częstotliwość odcięcia:
a) 211
b) 433
Rodzaj okna:
Hamming

5. Wyniki

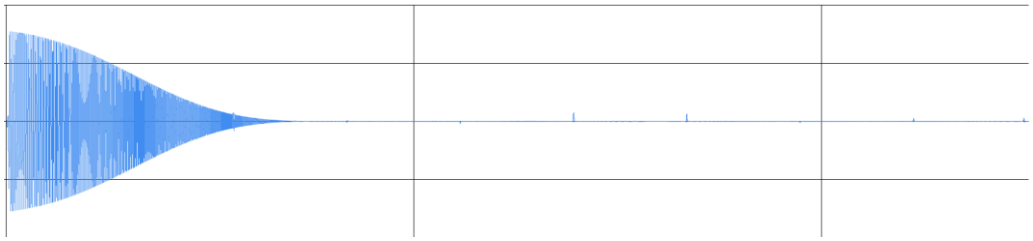
1. chirp_100Hz_1000Hz_lin.wav



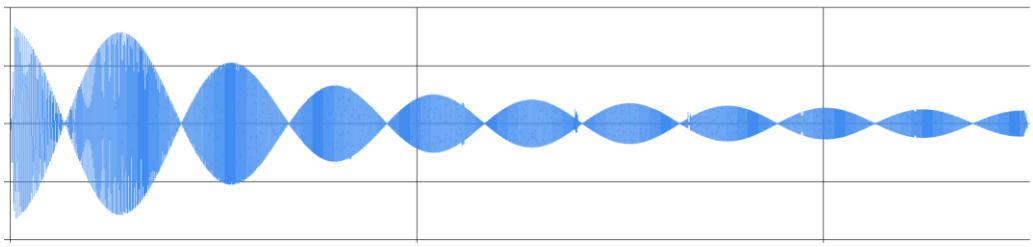
a)

dziedzina czasu:

przyczynowy
czas przetwarzania: 8,0 s

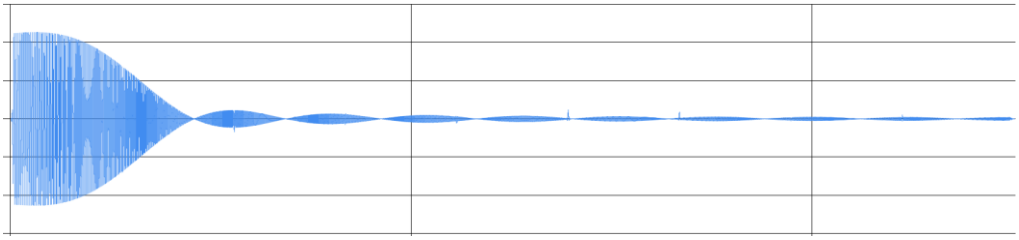


Nieprzyczynowy
czas przetwarzania: 8,53 s

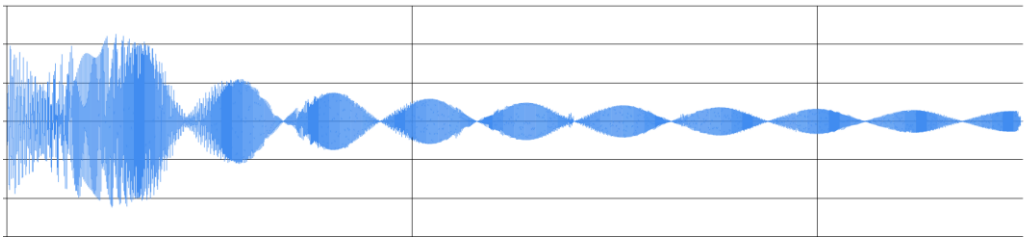


dziedzina częstotliwości:

przyczynowy
czas przetwarzania: 2,41 s



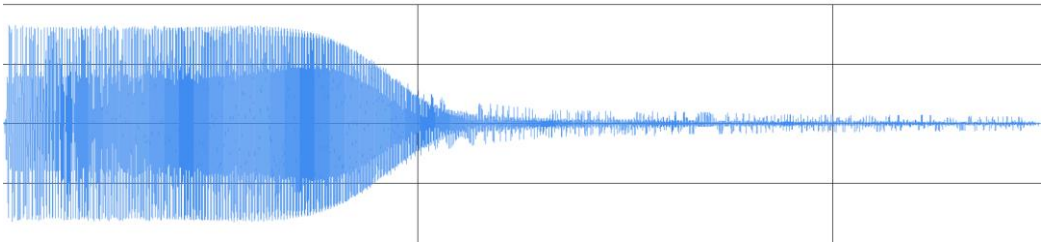
Nieprzyczynowy
czas przetwarzania: 2,37 s



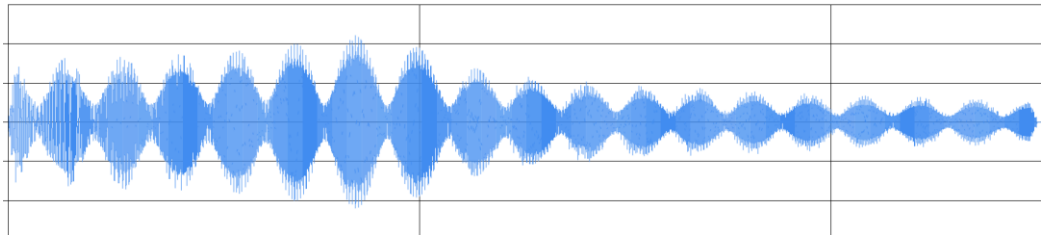
b)

dziedzina czasu:

przyczynowy
czas przetwarzania: 10,62 s

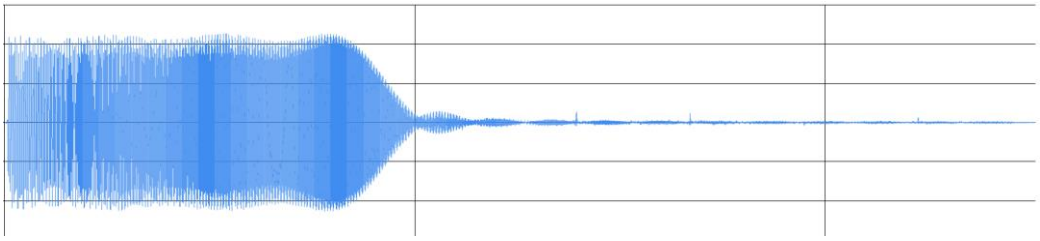


Nieprzyczynowy
czas przetwarzania:10,84 s

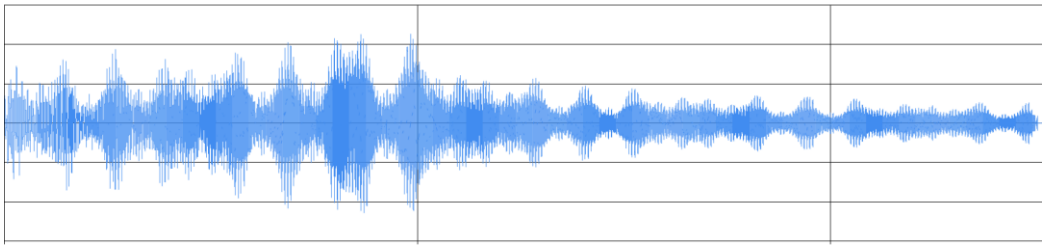


dziedzina częstotliwości:

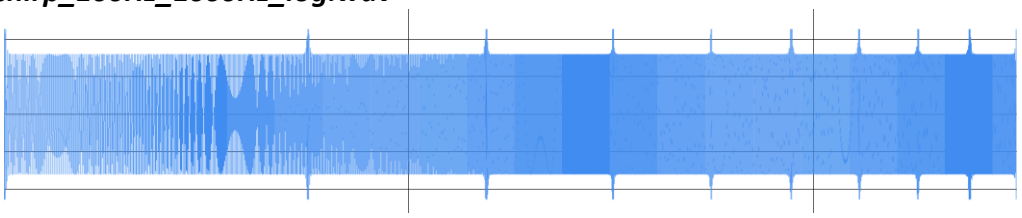
przyczynowy
czas przetwarzania: 2,24 s



Nieprzyczynowy
czas przetwarzania: 2,23 s



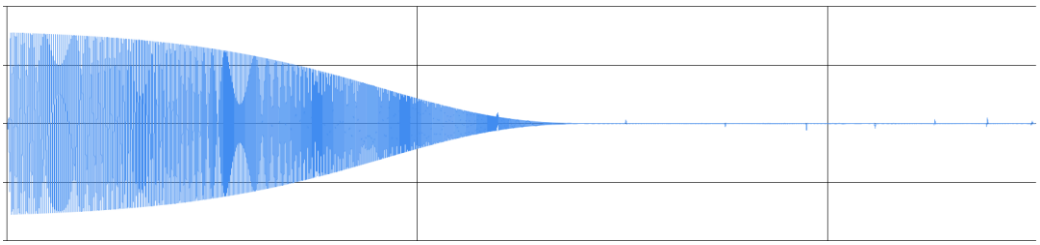
2. chirp_100Hz_1000Hz_log.wav



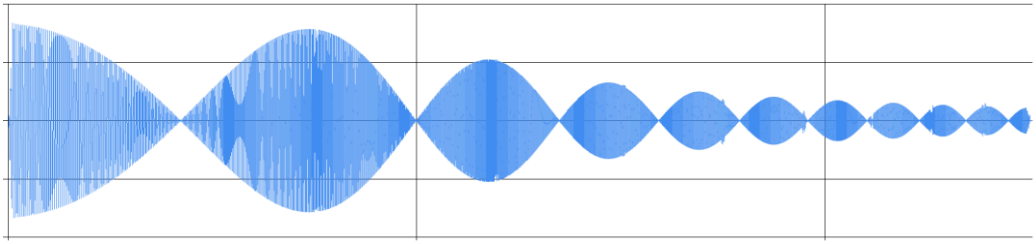
a)

dziedzina czasu:

przyczynowy
czas przetwarzania: 8,14 s

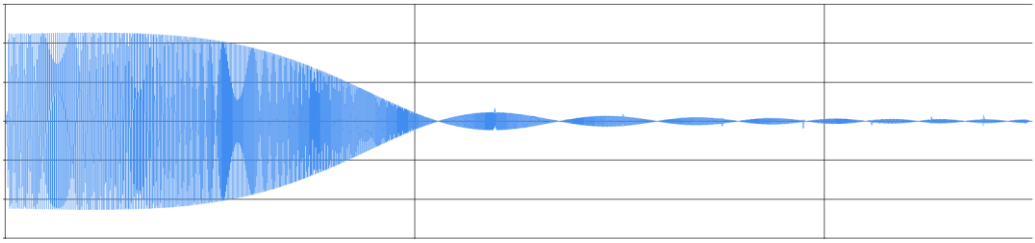


Nieprzyczynowy
Czas przetwarzania: 8,3 s

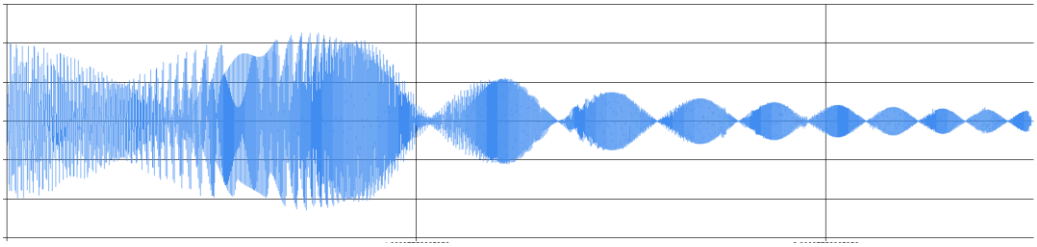


dziedzina częstotliwości:

przyczynowy
czas przetwarzania: 2,42 s



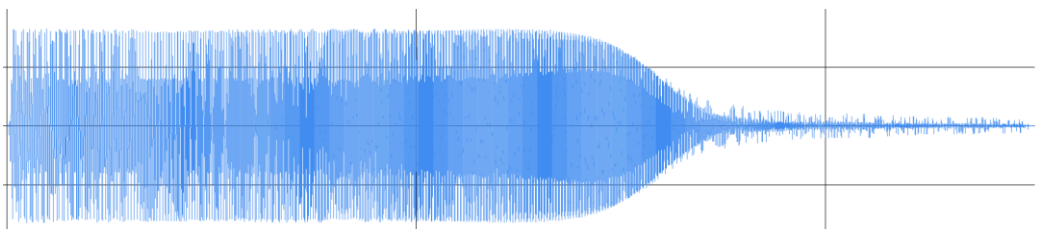
Nieprzyczynowy
Czas przetwarzania: 2,68 s



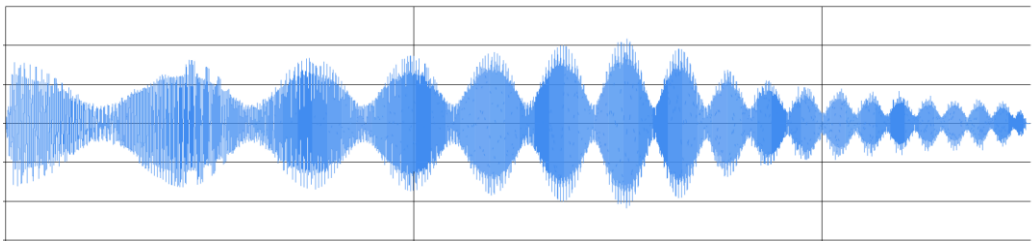
b)

dziedzina czasu:

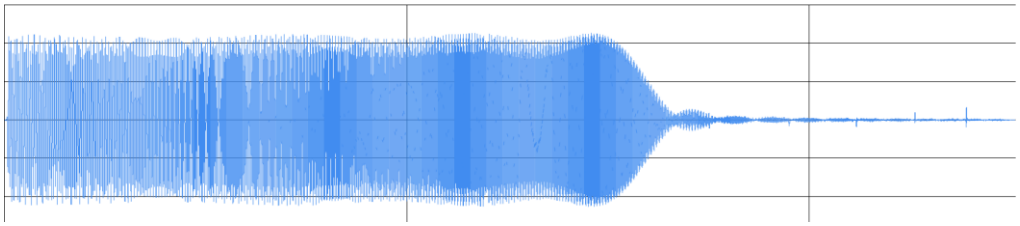
przyczynowy
czas przetwarzania: 10,67 s



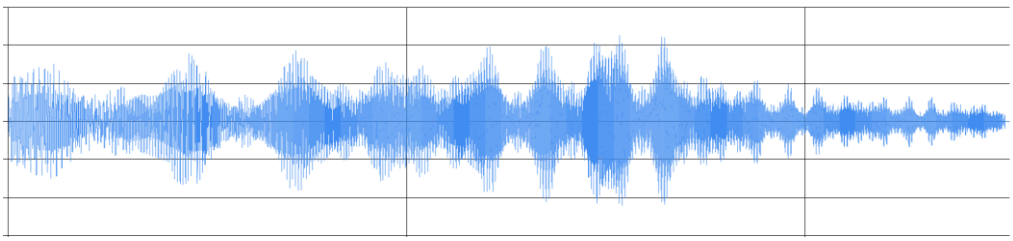
Nieprzyczynowy
czas przetwarzania: 10,67 s



dziedzina częstotliwości:
przyczynowy
czas przetwarzania: 2,01 s

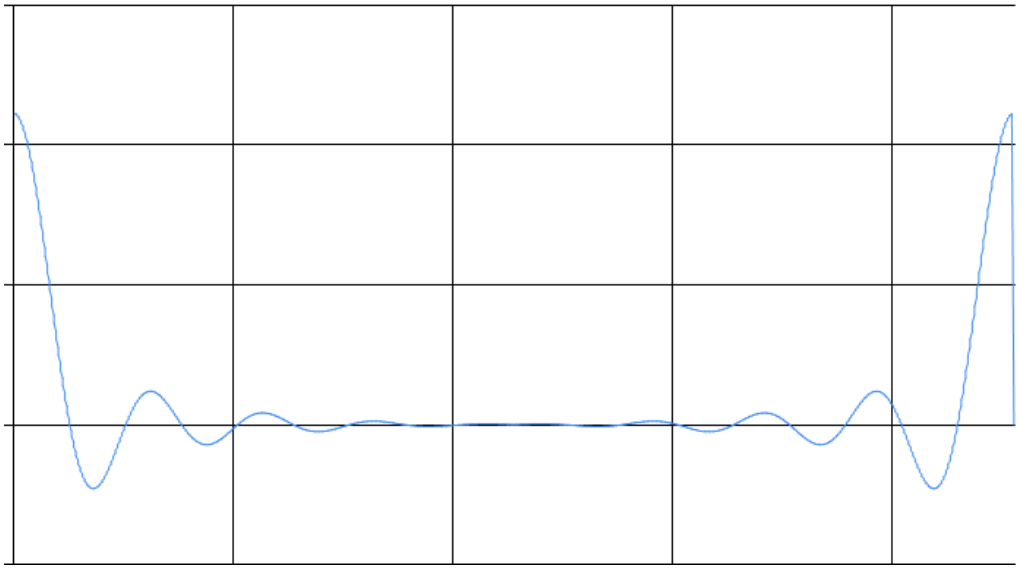


Nieprzyczynowy
czas przetwarzania: 2,22 s



6. Dyskusja

Otrzymane wyniki pokazują podobieństwo sygnałów wyjściowych mniej więcej do połowy długości sygnału. Druga część sygnału w dziedzinie częstotliwości jest zdecydowanie mniej wypłaszczona. Oczywiście wybrane parametry mają ogromne znaczenie dla otrzymywanych wyników. Możliwe jest zastosowanie w zasadzie dowolnej kombinacji parametrów co wskazuje na duże możliwości filtracji. Wyniki filtru nieprzyczynowego tworzą charakterystyczne powtórzenia sygnału z różnymi amplitudami, wpływa na to charakterystyka filtru nieprzyczynowego:



7. Wnioski

- szybkość filtracji w dziedzinie częstotliwości jest większa

- na wyjściu otrzymujemy sygnał zbliżony do sygnału wejściowego

- dla filtru nieprzyczynowego powstają charakterystyczne powtórzenia sygnału z co raz mniejszą amplitudą

- metoda okna jest prostym do zaimplementowania algorytmem filtracji sygnału dźwiękowego