Санкт-Петербургский Государственный Электротехнический Университет "ЛЭТИ"

кафедра физики

Задание №4 по дисциплине

"Физические основы информационных технологий"

Название: Фильтрация звукового сигнала.

|  |  |
| --- | --- |
| Фамилия И.О.: | Насонов Я.К. |
| группа: | 1303 |
| Преподаватель: | Альтмарк А.М. |
| Итоговый балл: |  |
|  |  |

Крайний срок сдачи: 05.12.23

.

Санкт-Петербург 2023

**Условие задания**

На входе приемника получен звуковой сигнал в двоичном коде (рис.1.). Необходимо перевести двоичный код в десятичный и затем провести над аналоговым сигналом процедуру фильтрации от высокочастотных помех. Для фильтрации необходимо использовать пассивные фильтры (фильтры без дополнительного источника питания), которые могут в себя включать, резисторы, конденсаторы и катушки индуктивности.

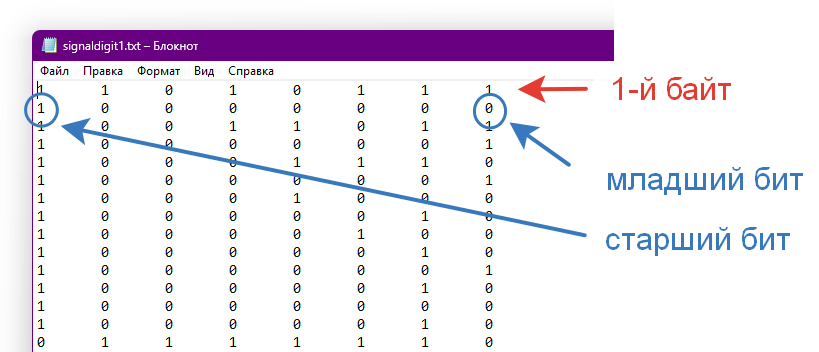


Рис.1. Структура данных в текстовом файле с сигналом

Исходные данные нужно взять в файле FOIT\_IDZ4.xlsx. В отчет нужно включить график сигнала во временной области и его спектр, схему фильтра и АЧХ его передаточной функции, спектр фильтрованного сигнала, а также график выходного сигнала во временной области. Файл IDZ4.txt должен содержать ответ на вопрос, который записан в звуком сигнале.

Помимо текстового файла IDZ4.txt в папке IDZ4 должен находиться Word-файл с отчетом, а также файл с кодом (Python, Mathcad, Mathematica). Для лучшего понимания отчетности смотрите папку “Пример организации яндекс-папки студентов”.

Пример содержания файла IDZ4.txt:

25

**Исходные данные (8 вариант)**

|  |  |
| --- | --- |
| длительность сигнала, с | Файл с сигналом |
| 3 | Signaldigit8.txt |

**Теоретические сведения**

Для получения частотной характеристики аналогового сигнала необходимо применить быстрое преобразование Фурье к аналоговому сигналу, на основе полученных данных можно построить спектр сигнала.

Фильтры могут использоваться для удаления высокочастотных помех из аналогового сигнала. Фильтры могут включать в себя резисторы (R), конденсаторы (C) и индуктивности (L).

Одним из таких фильтров является фильтр Баттерворта, его схема имеет следующий вид:

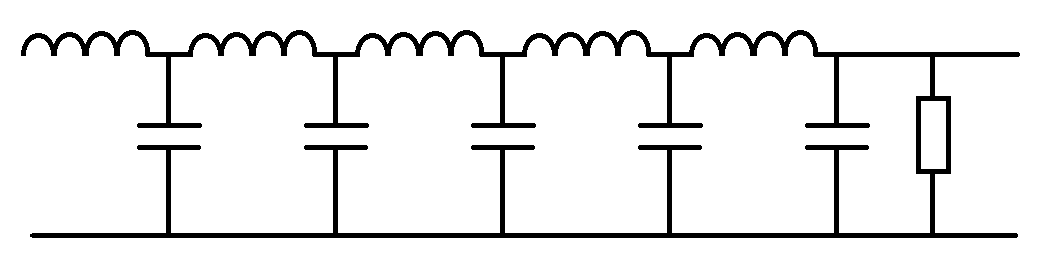


Рис.2 Схема фильтра

После задания параметров для данного фильтра и применения функции, описывающей передаточную функцию данного фильтра, можно найти спектр отфильтрованного сигнала.

Далее нужно произвести обратное преобразование Фурье отфильтрованного спектра и получить уже временной сигнал и в результате, с использованием специальной библиотеки, есть возможность произвести очищенный от помех сигнал.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Файл **IDZ4.py**

import numpy as np  
import matplotlib.pyplot as plt  
from scipy.fft import fft, ifft  
import sounddevice as sd  
  
  
def butterworth\_filter(w, Uin):  
 Z5 = 1 / (1j \* w \* C5 + 1 / R1)  
 Z4 = 1 / (1j \* w \* C4 + 1 / (1j \* w \* L5 + Z5))  
 Z3 = 1 / (1j \* w \* C3 + 1 / (1j \* w \* L4 + Z4))  
 Z2 = 1 / (1j \* w \* C2 + 1 / (1j \* w \* L3 + Z3))  
 Z = 1 / (1j \* w \* C1 + 1 / (1j \* w \* L2 + Z2))  
 I1 = Uin / (1j \* w \* L1 + Z)  
 U = I1 \* Z  
 I2 = U / (1j \* w \* L2 + Z2)  
 U2 = I2 \* Z2  
 I3 = U2 / (1j \* w \* L3 + Z3)  
 U3 = I3 \* Z3  
 I4 = U3 / (1j \* w \* L4 + Z4)  
 U4 = I4 \* Z4  
 I5 = U4 / (1j \* w \* L5 + Z5)  
 U5 = I5 \* Z5  
 Uout = U5  
 return Uout / Uin  
  
  
def show(plot1, plot2, title, xlabel):  
 plt.plot(plot1, plot2)  
 plt.title(title)  
 plt.xlabel(xlabel)  
 plt.ylabel('Amplitude')  
 plt.show()  
  
  
duration = 3  
  
filename = "./signaldigit8.txt"  
  
R1 = 50  
  
C1 = 0.000005  
C2 = 0.0000045  
C3 = 0.0000035  
C4 = 0.000002  
C5 = 497.9 \* 10 \*\* -9  
  
L1 = 0.01  
L2 = 0.012  
L3 = 0.01  
L4 = 0.007  
L5 = 0.003  
  
with open(filename, 'r') as file:  
 discrete\_signal = [line.strip().split('\t') for line in file]  
  
analog\_signal = [int(''.join(b), 2) for b in discrete\_signal]  
dt = duration / len(discrete\_signal)  
  
*# Входной сигнал*sample\_rate = 44100  
time\_axis = np.linspace(0, duration, len(analog\_signal))  
show(time\_axis, analog\_signal, 'Analog Signal', 'Time (s)')  
  
*# Спектр*fsig = fft(analog\_signal)  
out\_n = len(fsig)  
df = 1 / dt  
freq\_axis = np.fft.fftfreq(out\_n, dt)[1:]  
show(freq\_axis, np.abs(fsig[1:]), 'Spectrum', 'Frequency (Hz)')  
  
*# Частотная характеристика фильтра*Uin\_value = 1  
freq\_range = np.linspace(1, out\_n / 10, out\_n - 1)  
filter\_response = np.abs(butterworth\_filter(freq\_range, Uin\_value))  
show(freq\_range, filter\_response, 'Frequency response', 'Frequency (Hz)')  
  
*# Применение фильтра*H\_list = filter\_response  
F\_new = fsig[1:] \* H\_list  
show(freq\_axis, np.abs(F\_new), 'Filtered Spectrum', 'Frequency (Hz)')  
  
*# Обратное преобразование Фурье*changed\_signal = ifft(np.concatenate(([0], F\_new)))  
filtered\_time\_axis = np.linspace(0, duration, len(changed\_signal))  
show(filtered\_time\_axis, np.real(changed\_signal), 'Filtered Signal', 'Time (s)')  
  
*# Воспроизводим отфильтрованный сигнал*sd.play(np.real(changed\_signal), sample\_rate)  
sd.wait()