Санкт-Петербургский Государственный Электротехнический Университет "ЛЭТИ"

кафедра физики

Задание №4 по дисциплине

"Физические основы информационных технологий"

Название: Фильтрация звукового сигнала.

|  |  |
| --- | --- |
| Фамилия И.О.: | Иванов А. С. |
| 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.

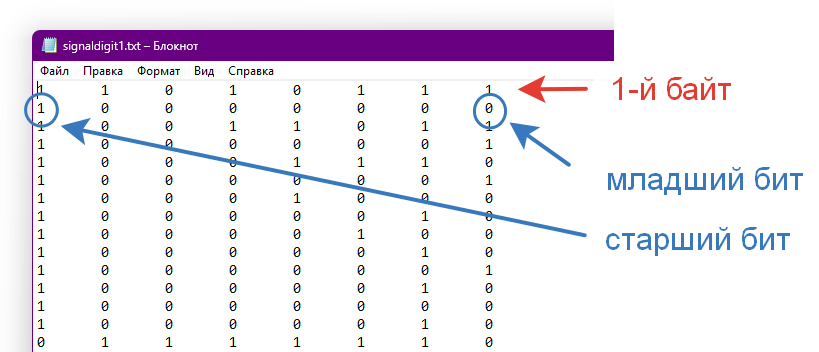


Рис.1. Структура данных в текстовом файле с сигналом

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 (5 вариант)**

|  |  |
| --- | --- |
| длительность сигнала, с | Файл с сигналом |
| 3 | signaldigit5.txt |

**Теоретические сведения**

Для анализа частотных характеристик аналогового сигнала применяется быстрое преобразование Фурье, что позволяет построить его спектр. Фильтры, такие как фильтр Баттерворта с использованием резисторов (R), конденсаторов (C) и индуктивностей (L), используются для удаления высокочастотных помех из сигнала.

После определения параметров фильтра Баттерворта и применения передаточной функции фильтра, мы можем анализировать спектр отфильтрованного сигнала. Затем, совершив обратное преобразование Фурье отфильтрованного спектра, получаем временной сигнал. С использованием специальной библиотеки можно далее создать очищенный от помех сигнал, что обеспечивает эффективную фильтрацию и анализ аналоговых сигналов.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

**Файл main.py**

import numpy as np  
import matplotlib.pyplot as plt  
from scipy.fft import fft, ifft  
import sounddevice as sd  
  
class SignalProcessor:  
 def \_\_init\_\_(self, filename):  
 self.filename = filename  
 self.discretCode = self.load\_signal()  
  
 def load\_signal(self):  
 with open(self.filename, 'r') as file:  
 discretCode = [line.strip().split('\t') for line in file]  
  
 return [int(''.join(b), 2) for b in discretCode]  
  
 def plot\_signal(self, time\_axis, signal, title, xlabel='Время', ylabel='Амплитуда'):  
 plt.plot(time\_axis, signal)  
 plt.title(title)  
 plt.xlabel(xlabel)  
 plt.ylabel(ylabel)  
 plt.show()  
  
 def filter\_butterworth(self, freq\_range, Uin, R1, L1, L2, L3, L4, L5, C1, C2, C3, C4, C5):  
 Z5 = 1 / (1j \* freq\_range \* C5 + 1 / R1)  
 Z4 = 1 / (1j \* freq\_range \* C4 + 1 / (1j \* freq\_range \* L5 + Z5))  
 Z3 = 1 / (1j \* freq\_range \* C3 + 1 / (1j \* freq\_range \* L4 + Z4))  
 Z2 = 1 / (1j \* freq\_range \* C2 + 1 / (1j \* freq\_range \* L3 + Z3))  
 Z = 1 / (1j \* freq\_range \* C1 + 1 / (1j \* freq\_range \* L2 + Z2))  
 I1 = Uin / (1j \* freq\_range \* L1 + Z)  
 U = I1 \* Z  
 I2 = U / (1j \* freq\_range \* L2 + Z2)  
 U2 = I2 \* Z2  
 I3 = U2 / (1j \* freq\_range \* L3 + Z3)  
 U3 = I3 \* Z3  
 I4 = U3 / (1j \* freq\_range \* L4 + Z4)  
 U4 = I4 \* Z4  
 I5 = U4 / (1j \* freq\_range \* L5 + Z5)  
 U5 = I5 \* Z5  
 Uout = U5  
 return Uout / Uin  
  
 def process\_signal(self, R1, L1, L2, L3, L4, L5, C1, C2, C3, C4, C5, sample\_rate=45000):  
 dt = 3 / len(self.discretCode)  
 time\_axis = np.linspace(0, 3, len(self.discretCode))  
  
 self.plot\_signal(time\_axis, self.discretCode, 'Сигнал во временной области')  
  
 fsig = fft(self.discretCode)  
 out\_n = len(fsig)  
 freq\_axis = np.fft.fftfreq(out\_n, dt)[1:]  
  
 self.plot\_signal(freq\_axis, np.abs(fsig[1:]), 'Спектр сигнала во временной области', 'Частота', 'Амплитуда')  
  
 Uin\_value = 1  
 freq\_range = np.linspace(1, out\_n / 10, out\_n - 1)  
 filter\_response = np.abs(self.filter\_butterworth(freq\_range, Uin\_value, R1, L1, L2, L3, L4, L5, C1, C2, C3, C4, C5))  
  
 self.plot\_signal(freq\_range, filter\_response, 'АЧХ', 'Частота', 'Амплитуда')  
  
 H\_list = filter\_response  
 F\_new = fsig[1:] \* H\_list  
  
 self.plot\_signal(freq\_axis, np.abs(F\_new), 'Спектр фильтрованного сигнала', 'Частота', 'Амплитуда')  
  
 changed\_signal = ifft(np.concatenate(([0], F\_new)))  
 filtered\_time\_axis = np.linspace(0, 3, len(changed\_signal))  
  
 self.plot\_signal(filtered\_time\_axis, np.real(changed\_signal), 'Выходной сигнал временной области')  
  
 sd.play(np.real(changed\_signal), sample\_rate)  
 sd.wait()  
  
# Параметры фильтра и обработки сигнала  
R1 = 50  
L1 = 0.01  
L2 = 0.012  
L3 = 0.01  
L4 = 0.007  
L5 = 0.003  
C1 = 0.000005  
C2 = 0.0000045  
C3 = 0.0000035  
C4 = 0.000002  
C5 = 497.9 \* 10 \*\* -9  
  
filename = "./signaldigit5.txt"  
  
# Создание экземпляра SignalProcessor и обработка сигнала  
signal\_processor = SignalProcessor(filename)  
signal\_processor.process\_signal(R1, L1, L2, L3, L4, L5, C1, C2, C3, C4, C5)