Санкт-Петербургский Государственный Электротехнический
Университет "ЛЭТИ"

кафедра физики

Задание №4 по дисциплине

"Физические основы информационных технологий"

Название: Фильтрация звукового сигнала

Фамилия И.О.:	Чернякова В.А.
Группа:	1304
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.

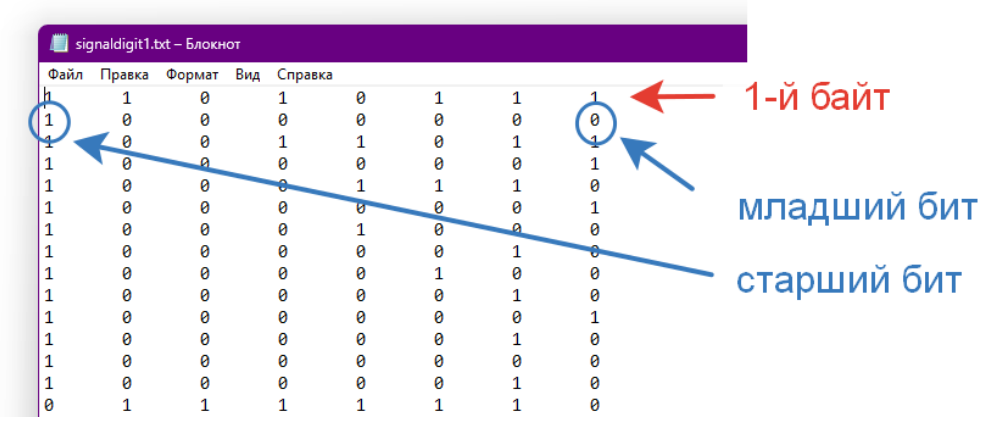

Рис.1. Структура данных в текстовом файле с сигналом

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.

Вариант 18.

Данные.

Длительность сигнала, с: 3.75

Файл с сигналом: signaldigit18.txt

Основные теоретические положения.

Для построения спектра сигнала, заданного списком значений, используется дискретное преобразование Фурье:

$F_i = \sum_{k=1}^{N} S_k \exp\left(-j\,\frac{2\pi}{N}\,in\right)$

Абсолютное значение |F| и будет АЧХ спектра сигнала. В нем, как следует из названия, есть шкалы амплитуды и частоты. АЧХ спектра

гармонического колебания будет представлена одной чертой, а ω - координата будет соответствовать ее частоте в исходном сигнале.

Для того, чтобы очистить сигнал от помех, используется фильтр. В данной лабораторной работе используется именно фильтр Баттерворта.

На рисунке 2 показана схема фильтра.

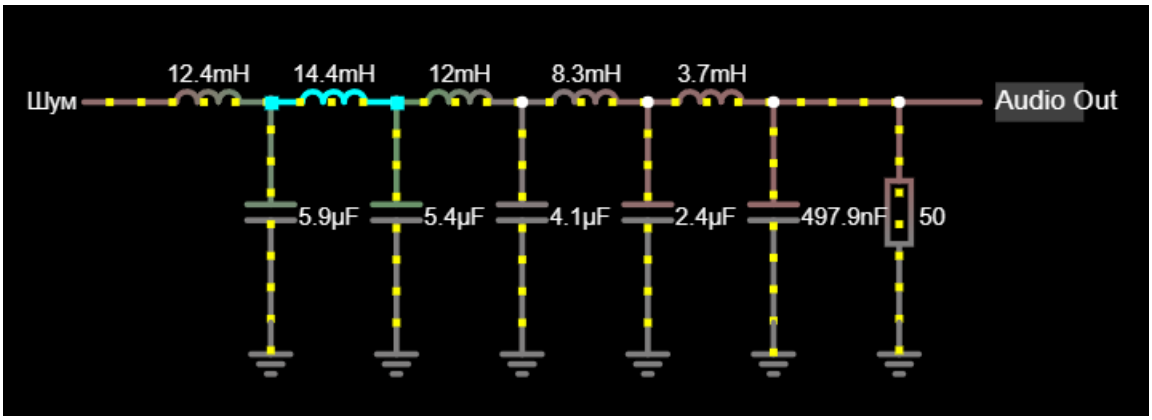

Рисунок 2 – фильтр Баттерворта.

АЧХ его передаточной функции имеет вид ступеньки, что позволяет достаточно точно обрезать высокие частоты шума, почти не ослабив основной сигнал. Схема фильтра представлена на рисунке.

Передаточная функция системы H = $\frac{Uout}{Uin}$

Она показывает связь между входным и выходным напряжениями.

Для нахождения входного и выходного напряжения нужно знать импеданс Z – комплексное сопротивление для гармонического сигнала:

Резистор	Z_R	R
Конденсатор	Z_C	$\frac{1}{j\omega C}$
Катушка индуктивности	Z_L	jωL

АЧХ передаточной функции получается как |H(jω)|, и показывает изменение амплитуды сигнала в зависимости от его частоты.

Выполнение работы.

Были объявлены переменные, в которых хранятся данные, соответствующие варианту задания.

Преобразование бинарных чисел, считанных из файла, в десятичные числа. Используется функция FromDigits для каждого элемента

signalBinaryCode и осуществляется конвертация из двоичной системы в десятичную.

$$\text{Table[FromDigits[signalBinaryCode[[}i\text{]],2], \{}i\text{, 1, Length@signalBinaryCode\}];}$$

Воспроизведение аналогового сигнала. ListPlay в Mathematica используется для воспроизведения звука на основе списка амплитудных значений. В данном случае, signalAnalog представляет собой список десятичных чисел, который интерпретируется как амплитуды звукового сигнала. Параметр SampleRate -> 44100 указывает на частоту дискретизации звука в герцах.

$$\text{ListPlay[signalAnalog, SampleRate}-> 44100\text{]}$$

Создание списка пар значений {время, амплитуда}. Значение времени вычисляется как (i - 1) * dt, где i - порядковый номер элемента в списке, а dt - шаг по времени между элементами. Амплитуда берется из списка signalAnalog.

$$\text{Table[\{}(i-1)*\text{dt, signalAnalog[[}i\text{]]\}, \{}i\text{, 1, Length@signalBinaryCode\}]}$$

Применяем преобразование Фурье к аналоговому сигналу signalAnalog.

$$\text{signalFourier} = \text{Fourier[signalAnalog];}$$

Переменная outN, которая содержит количество элементов в полученном массиве signalFourier.

$$\text{outN} = \text{Length@signalFourier;}$$

Получение частоты.

$$\text{df} = 1/t;$$

Создание нового списка, представляющего амплитуды частот в результате преобразования Фурье. Каждый элемент этого списка представляет собой пару значений: первое значение - частота (вычисленная как 2 π df (i - 1)), второе значение - амплитуда (вычисленная как абсолютное значение Abs@signalFourier[[i]]). Здесь используется i от 2 до outN, так как нулевой элемент в signalFourier содержит постоянную компоненту.

$$\text{Table[\{}2\pi\text{df}(i-1)\text{, Abs@signalFourier[[}i\text{]]\}, \{}i\text{, 2, outN\}]}$$

Далее для фильтрации применялся фильтр Баттерворта. Значения сопротивления, емкостей и индуктивностей взяты с сайта falstad.

Определение импедансов каждой части цепи.

Zpar5[ω], Zpar4[ω], Zpar3[ω], Zpar2[ω], Zpar[ω]: Импедансы пятой, четвертой, третьей, второй и первой части цепи соответственно.

Определение токов в каждой части цепи.

I1[ω], I2[ω], I3[ω], I4[ω], I5[ω]: Токи в первой, второй, третьей, четвертой и пятой части цепи соответственно.

Определение напряжений на компонентах каждой части цепи.

Upar1[ω], Upar2[ω], Upar3[ω], Upar4[ω], Upar5[ω]: Напряжения на компонентах первой, второй, третьей, четвертой и пятой части цепи соответственно.

Определение передаточной функции и фильтрация.

Uout[ω]: Выходное напряжение.

H[ω]: Передаточная функция.

Амплитуды передаточной функции для каждой частоты в диапазоне.

$$\mathrm{Table}[\mathrm{Abs}@H[i], \{i, 1, \mathrm{outN}\}]$$

Фильтрация сигнала. Умножение исходного сигнала signalFourier на амплитуды передаточной функции.

$$\mathrm{signalFourier} * \mathrm{Hlist}$$

Создание таблицы значений, для построения спектра после фильтрации. {2 π df (i - 1), Abs@signalFourierFiltered[[i]]}: Это выражение, которое формирует пару значений для каждого i. Первое значение в паре - это частота, рассчитанная как 2 π df (i - 1), а второе значение - это амплитуда, вычисленная как абсолютное значение Abs@signalFourierFiltered[[i]]. {i, 2, outN}: это часть Table, определяющая, по какому диапазону i происходит итерация.

$$\mathrm{Table}\big[\{2\pi \mathrm{df}(i-1), \mathrm{Abs@signalFourierFiltered}\big[[i]\big]\}, \{i, 2, \mathrm{outN}\}\big];$$

Обратное преобразование Фурье для полученного фильтрованного сигнала.

$$\mathrm{InverseFourier[signalFourierFiltered]};$$

Создание таблицы значений, для построения графика сигнала после фильтрации. {(i - 1) *dt, Re@signalFiltered[[i]]}: это выражение формирует пару значений для каждого i. Первое значение в паре — это время, рассчитанное как (i - 1) *dt, где dt - шаг по времени между элементами. Второе

значение — это вещественная часть (Re) фильтрованного сигнала для соответствующего значения i.

$$\text{Table}\left[\left\{(i-1)*\text{dt}, \text{Re@signalFiltered}\left[[i]\right]\right\}, \{i, 1, \text{Length@signalBinaryCode}\}\right];$$

Разработанный программный код смотри в приложении А.

Тестирование.

На рисунках 3 – 7 представлены результат работы программы.

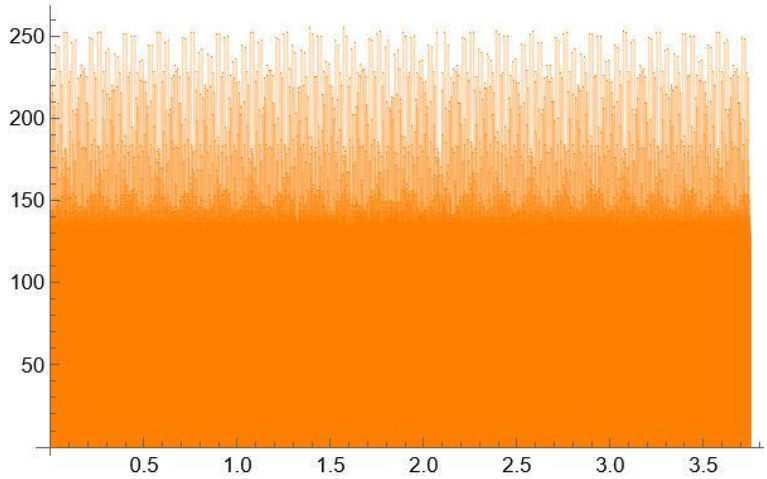

Рисунок 3 – график сигнала без фильтрации.

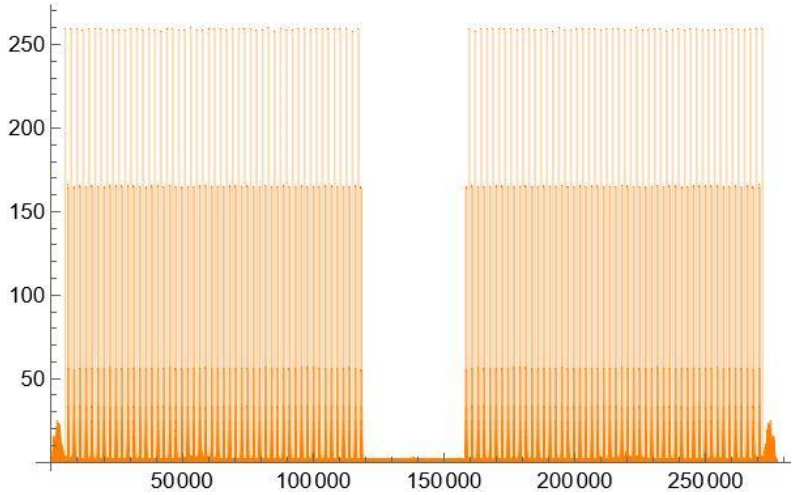

Рисунок 4 – спектр сигнала без фильтрации.

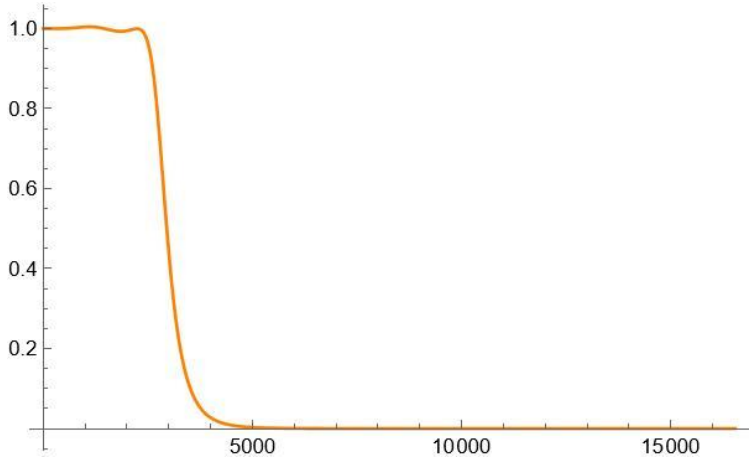

Рисунок 5 – АЧХ передаточной функции фильтра.

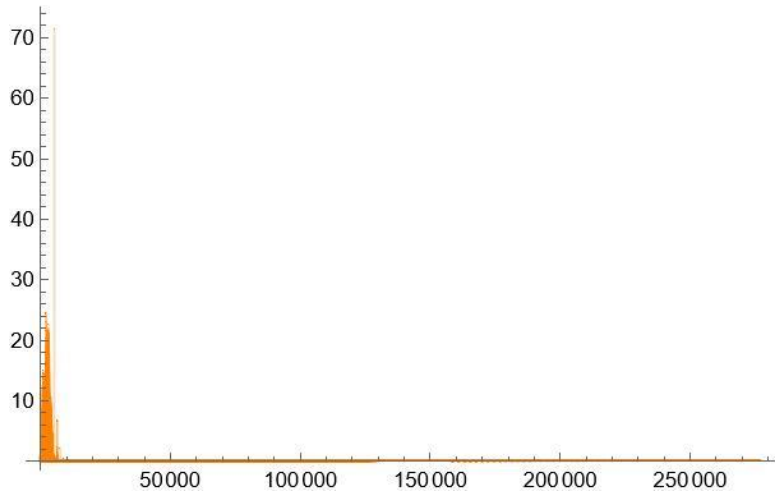

Рисунок 6 – спектр фильтрованного сигнала.

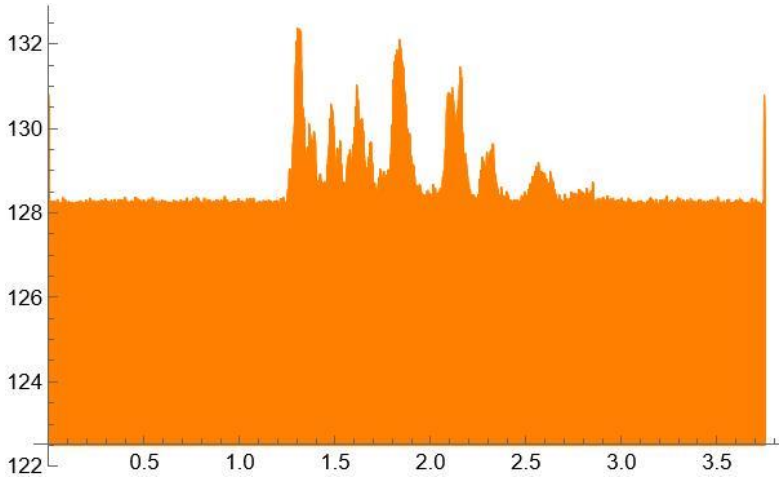

Рисунок 7 – график фильтрованного сигнала.

Выводы.

В ходе лабораторной работы написана программа, которая фильтрует входной сигнал, а также строит графики: сигнала без фильтрации, спектр сигнала без фильтрации, АЧХ передаточной функции фильтра Баттерворта, спектр фильтрованного сигнала и его график.

ПРИЛОЖЕНИЕ А.
ИСХОДНЫЙ КОД ПРОГРАММЫ

Файл: IDZ4.nb

```
signalBinaryCode = ReadList[
"C:\\Users\\22153\\Documents\\leti\\foit\\IDZ4\\signal\\signaldigit18.txt",
{Number, Number, Number, Number, Number, Number,
    Number, Number}];
signalAnalog =
  Table[FromDigits[signalBinaryCode[[i]], 2], {i, 1,
    Length@signalBinaryCode}];
ListPlay[signalAnalog, SampleRate -> 44100]
t = 3.75;
dt = t / Length@signalBinaryCode;
signalPlot =
  Table[{(i - 1) * dt, signalAnalog[[i]]}, {i, 1,
    Length@signalBinaryCode}];
ListPlot[signalPlot, Filling -> Axis, PlotRange -> Full,
 PlotStyle -> Orange]
signalFourier = Fourier[signalAnalog];
df = 1 / t;
outN = Length@signalFourier;
signalFourierAbs  =
  Table[{2 \[Pi] df (i - 1), Abs@signalFourier[[i]]}, {i, 2, outN}];
ListPlot[signalFourierAbs, Filling -> Axis, PlotRange -> Full,
 PlotStyle -> Orange]
coef = 2.25;
R = 50;
C1 = 5.9 * 10^-6 * coef;
C2 = 5.4 * 10^-6 * coef;
C3 = 4.1 * 10^-6 * coef;
C4 = 2.4 * 10^-6 * coef;
C5 = 497.9 * 10^-9 * coef;
L1 = 12.4 * 10^-3 * coef;
L2 = 14.4 * 10^-3 * coef;
L3 = 12 * 10^-3 * coef;
L4 = 8.3 * 10^-3 * coef;
L5 = 3.7 * 10^-3 * coef;
Zpar5[\[Omega]_] = 1/(I \[Omega] C5 + 1/R);
Zpar4[\[Omega]_] =
  1/(I \[Omega] C4 + 1/(I \[Omega] L5 + Zpar5[\[Omega]]));
Zpar3[\[Omega]_] =
  1/(I \[Omega] C3 + 1/(I \[Omega] L4 + Zpar4[\[Omega]]));
Zpar2[\[Omega]_] =
```

```
  1/(I \[Omega] C2 + 1/(I \[Omega] L3 + Zpar3[\[Omega]]));
Zpar[\[Omega]_] =
  1/(I \[Omega] C1 + 1/(I \[Omega] L2 + Zpar2[\[Omega]]));
I1[\[Omega]_] = Uin/(I \[Omega] L1 + Zpar[\[Omega]]);
Upar1[\[Omega]_] = I1[\[Omega]]*Zpar[\[Omega]];
I2[\[Omega]_] = Upar1[\[Omega]]/(I \[Omega] L2 + Zpar2[\[Omega]]);
Upar2[\[Omega]_] = I2[\[Omega]] * Zpar2[\[Omega]];
I3[\[Omega]_] = Upar2[\[Omega]]/(I \[Omega] L3 + Zpar3[\[Omega]]);
Upar3[\[Omega]_] = I3[\[Omega]] * Zpar3[\[Omega]];
I4[\[Omega]_] = Upar3[\[Omega]]/(I \[Omega] L4 + Zpar4[\[Omega]]);
Upar4[\[Omega]_] = I4[\[Omega]] * Zpar4[\[Omega]];
I5[\[Omega]_] = Upar4[\[Omega]]/(I \[Omega] L5 + Zpar5[\[Omega]]);
Upar5[\[Omega]_] = I5[\[Omega]] * Zpar5[\[Omega]];
Uout[\[Omega]_] = Upar5[\[Omega]];
H[\[Omega]_] = Uout[\[Omega]]/Uin;
Hlist = Table[Abs@H[i], {i, 1, outN}];
signalFourierFiltered = signalFourier*Hlist;
Plot[Abs@H[\[Omega]], {\[Omega], 1, outN/10}, PlotRange -> Full,
 PlotStyle -> Orange]
signalFourierFilteredTable =
  Table[{2 \[Pi] df (i - 1), Abs@signalFourierFiltered[[i]]}, {i, 2,
    outN}];
ListPlot[signalFourierFilteredTable , Filling -> Axis,
 PlotRange -> Full, PlotStyle -> Orange]
signalFiltered  = InverseFourier[signalFourierFiltered];
signalFilteredTable =
  Table[{( i - 1)*dt, Re@signalFiltered[[i]]}, {i, 1,
    Length@signalBinaryCode}];
ListPlot[signalFilteredTable, Filling -> Axis, PlotRange -> Full,
 PlotStyle -> Orange]
ListPlay[Re@signalFiltered , SampleRate -> 44100]
```