УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

Департман за рачунарство и аутоматику
Смер рачунарство и аутоматика
Одсек за рачунарску технику и рачунарске комуникације

ПРОЈЕКАТ

Кандидат: **Лазар Јовановић**
Број индекса: **RA 14/2020**

Предмет: **Алгоритми дигиталне обраде звука**
Тема рада: **Обрада звучних сигнала**

Нови Сад, децембар 2022.

УВОД

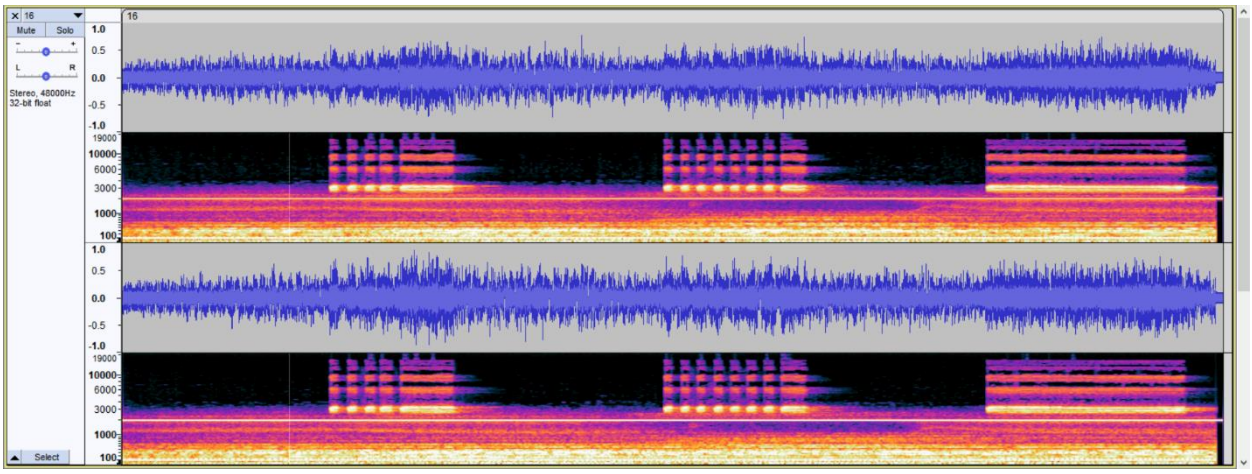
Циљ овогодишњег пројекта био је да, уз помоћ алата: Audacity, WinFilter i Code Composer Studio, и користећи знање стечено на вежбама у току семестра, реализујемо систем чија је улога филтрирање улазног сигнала, по спецификацији пројекта, тако да се на излазу добије само жељена компонента сигнала.

Конкретно, решење овог проблема добио сам решавањем четири постављена задатка, користећи одговарајуће филтере, тако да се од улазног сигнала састављеног од звука пиштаљке, звука хора и два синуса, добије излазни сигнал у коме се чује само звук пиштаљке.

ЗАДАТАК 1

Први корак у решавању пројектног задатка био је одређивање жељених и нежељених фреквенција у сигналу. Овај проблем решава се коришћењем Audacity алата.

Додавањем сигнала бр. 16 у Audacity и репродуковањем истог, може се чути звук пиштаљке, хора и звук синусног сигнала. Коришћењем опције Multiview можемо видети синусни облик овог сигнала као и његов спектограм (сл1).



слика 1

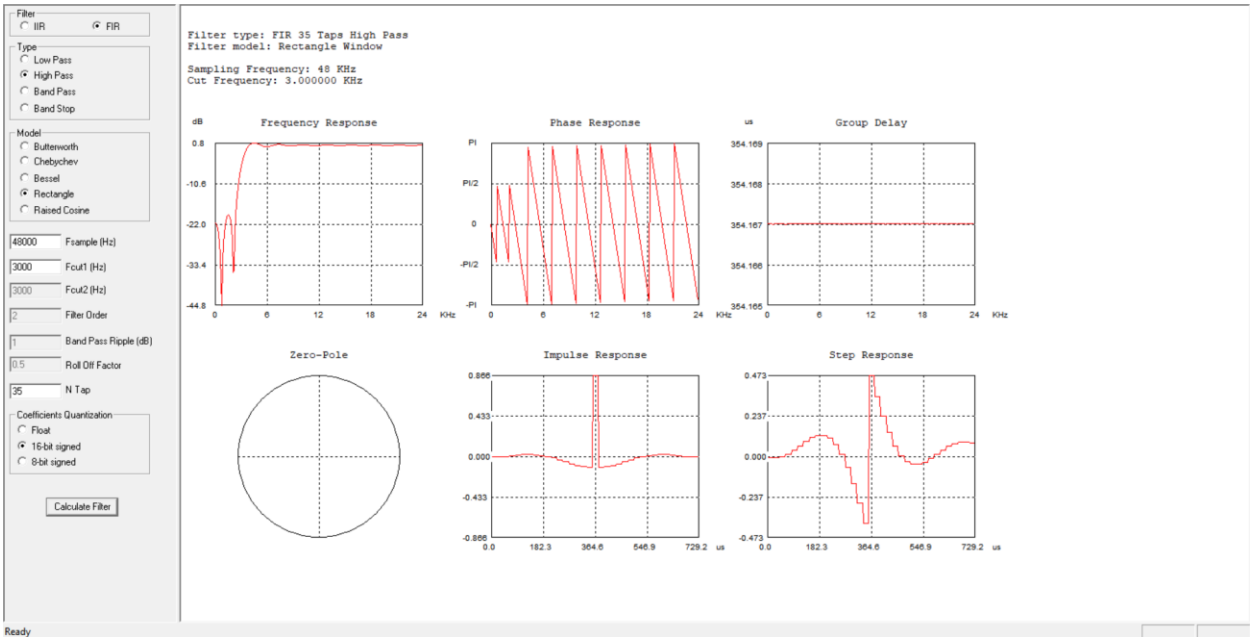
На основу спектограма може се видети опсег фреквенција улазног сигнала: [0 – 15000Hz]. Подешавањем величине прозора у подешавањима спектограма можемо одредити и опсег фреквенција нама корисног сигнала [~2500 – 15000Hz]. Нама непожељни сигнали се налазе у нижим опсезима фреквенција а фреквенција синуса је ~2000Hz.

Овим смо одредили фреквенцијски опсег нама корисног сигнала и тражену фреквенцију синусног сигнала као и да уколико желимо да издвојимо исти, морамо потиснути сигнале на фреквенцијама нижим од 2500Hz.

ЗАДАТАК 2

Након што смо успешно закључили опсег фреквенција нама корисног сигнала, као и фреквенцију синусног сигнала, можемо прећи на решавање другог задатка.

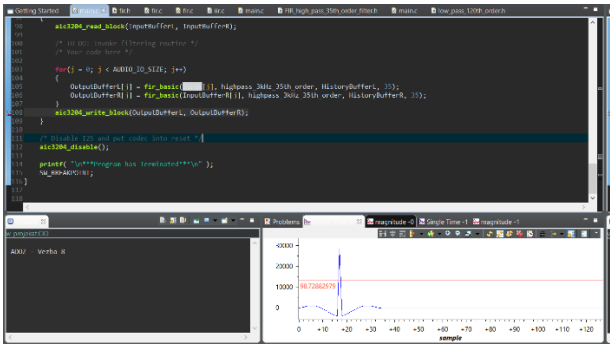
Користећи алат WinFilter, а пратећи упутства задатка, генеришу се коефицијенти за високо-пропусни филтар “35-ог реда”, где је редом дефинисан број коефицијената који описују филтер. Доњу границу унећемо као fcut1 u WinFilteru чиме ћемо добити коефицијенте филтра којим се потиска непожељни сигнал (сл. 2).



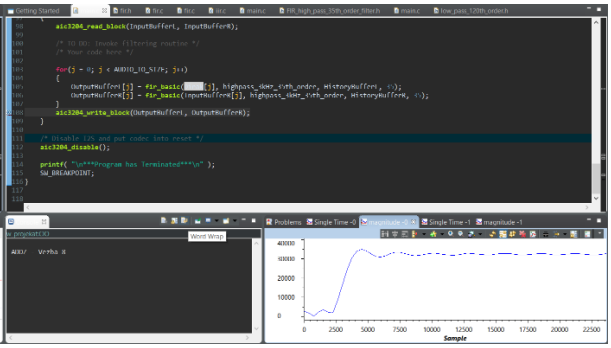
слика 2

У Code Composer-у генеришемо FIR_highpass_filters.h заглавље у које уврштавамо коефицијенте добијене раније. Одавде их прослеђујемо као параметре fir_basic функције у којој је имплементирана логика филтра.

Након успешне имплементације функције унутар Code Composer-а генеришемо сигнале помоћи опција Tools->Graph->Single Time (сл. 3), као и Tools->Graph->FFT Magnitude (сл. 4).

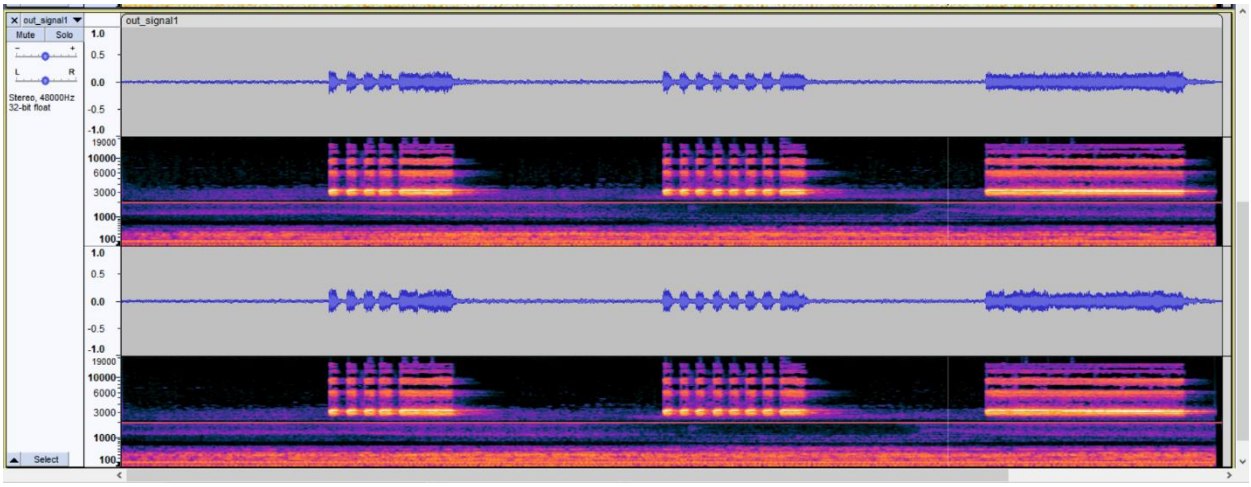


слика 3



слика 4

Након извршавања нашег кода добијамо излазни сигнал који убацујемо у Audacity и поредимо сигнал пре и после филтрирања (сл. 5).



слика 5

Може се видети да је део сигнала на фреквенцијама испод 2000Hz потиснут, али не и уклоњен. То је последица ниског реда филтра, као и имперфекција самог филтра генерисаног помоћу WinFiltera.

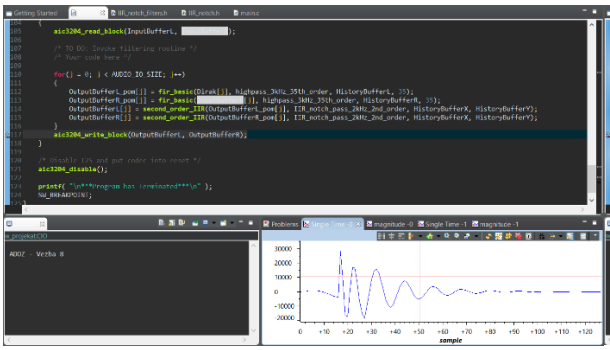
ЗАДАТАК 3

Слушањем добијеног излазног сигнала могуће је приметити да синусни сигнал није потиснут веома добро. То је последица имперфекција филтера као и чињеница да се синусни сигнал налази јако близу доње границе филтра. Да би решили овај проблем, користимо Notch филтар.

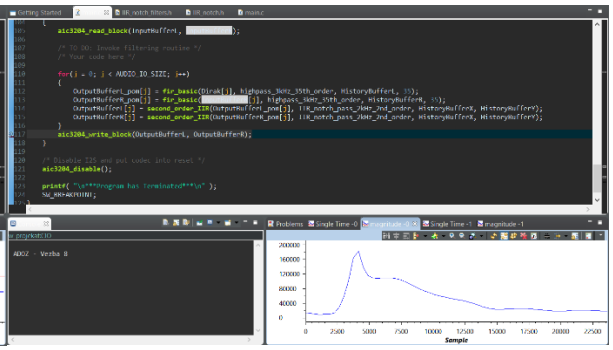
Користећи формулу дату у поставци задатка, коефицијенте Notch филтра рачунамо на следећи начин: a0=1, a1=-cos(2000/48000*2*pi), a2=1, b0=1, b1=-0.97*cos(2000/48000*2*pi), b2=1, где су коефицијенти a1 и b1 у старту подељени са 2 и услед Int16 имплементације сваки од ових коефицијената помножен са 32767 како би се вредности правилно скалирале. За константу r узето је 0.97.

Аналогно претходном задатку, правимо заглавље IIR_notch_filters.h у које смештамо вредности коефицијената. Одатле их прослеђујемо као параметре функцији second_order_IIR у којој је имплементирана логика филтра,

Након успешне имплементације унутар Code Composer-a генеришемо сигнале помоћу опција Tools->Graph->Single Time (сл. 6) и Tools->Graph->FFT Magnitude (сл. 7).

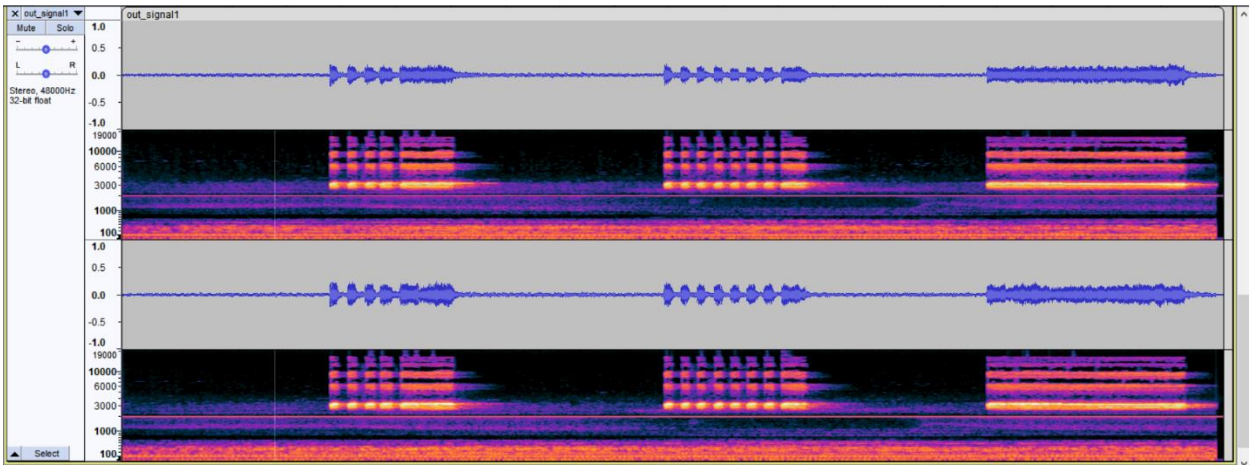


слика 6



слика 7

Након извршења кода, добијени сигнал убацујемо у Audacity и закључујемо да је синусни сигнал успешно потиснут (сл. 8).



слика 8

ЗАДАТАК 4

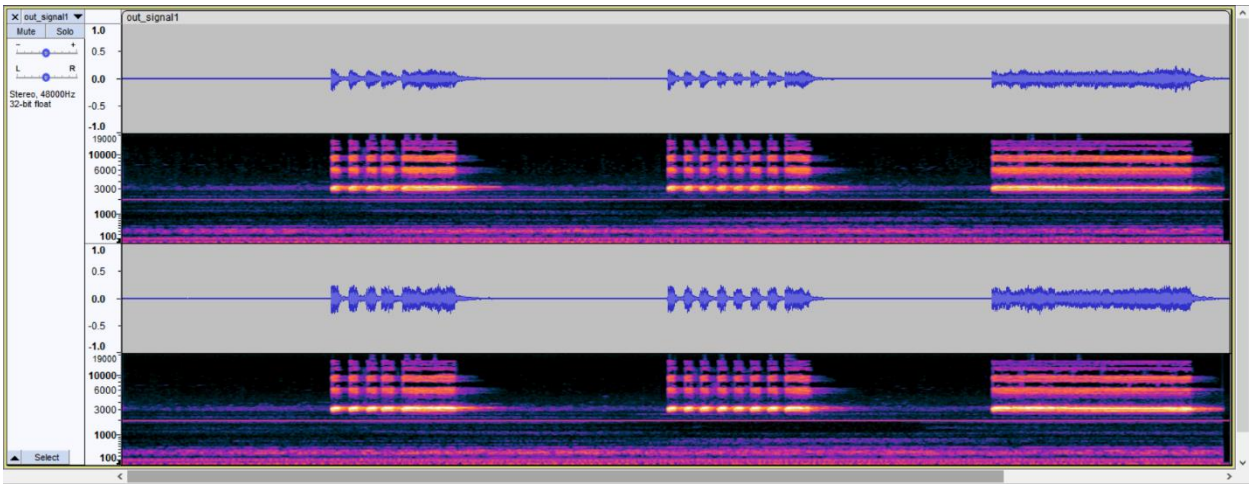
Након успешно одрађених претходних задатака, очекујемо да ћемо у излазном сигналу добити само жељени сигнал. Међутим то није случај. Приликом репродукције сигнала и даље су приметни утицаји нежељених сигнала. За ово је крива недовољна прецизност филтера, те се намеће логично решење овог проблема. У WinFilter алату генеришемо FIR филтре 77. и 129. реда.

Сличан проблем се јавља и код синусног сигнала, али и слично решње. Да би повећали прецизност филтрирања сигнал ћемо филтрирати 2 односно 3 пута како би симулирали ефекте IIR филтара 4. односно 6. реда.

За филтер 4. реда користили смо функцију fourth_order_IIR, док смо за филтер 6. реда имплементирали функцију sixth_order_IIR.

Поново понављамо поступак из ранијих задатака: Tools->Graph->Single Time и Tools->Graph->FFT Magnitude, како бисмо видели ефекат имплементираних филтера.

Такође, понављамо поступак и у Audacity-ју и упоређујемо ново добијени излаз са оригиналним, и излазима за филтере нижих редова (сл. 9).



слика 9

ЗАКЉУЧАК

Након успешне реализације пројекта успели смо да уз помоћ задатих алата, стеченог знања и уложеног напора направимо систем који издваја жељену компоненту из улазног сигнала уз минимално сметњи и нежељених сигнала.