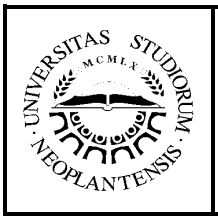


УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
НОВИ САД
Департман за рачунарство и аутоматику
Одсек за рачунарску технику и рачунарске комуникације

ЗАВРШНИ (BACHELOR) РАД

Кандидат: **Јована Јездимировић**
Број индекса: **РА190/2018**

Тема рада: **Реализација кодера за пренос рендеринг коефицијената заснованог на утискивању метаподатака у аудио податке на ДСП**

Ментор рада: **Доц. др Јелена Ковачевић**

Нови Сад, септембар, 2022

УНИВЕРЗИТЕТ У НОВОМ САДУ **ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА**
21000 НОВИ САД, Трг Доситеја Обрадовића 6

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:		
Идентификациони број, **ИБР**:		
Тип документације, **ТД**:		Монографска документација
Тип записа, **ТЗ**:		Текстуални штампани материјал
Врста рада, **ВР**:		Завршни (Bachelor) рад
Аутор, **АУ**:		Јована Јездимировић
Ментор, **МН**:		доц. др Јелена Ковачевић
Наслов рада, **НР**:		Реализација кодера за пренос рендеринг коефицијената заснованог на утискивању метаподатака у аудио податке на ДСП
Језик публикације, **ЈП**:		Српски / ћирилица
Језик извода, **ЈИ**:		Српски
Земља публиковања, **ЗП**:		Република Србија
Уже географско подручје, **УГП**:		Војводина
Година, **ГО**:		2022
Издавач, **ИЗ**:		Ауторски репринт
Место и адреса, **МА**:		Нови Сад; трг Доситеја Обрадовића 6
Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога)		6/34/15/4/23/0/0
Научна област, **НО**:		Електротехника и рачунарство
Научна дисциплина, **НД**:		Рачунарска техника
Предметна одредница/Кључне речи, **ПО**:		Дигитална обрада сигнала, рендеринг, стеганографија
УДК		
Чува се, **ЧУ**:		У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, **ВН**:		
Извод, **ИЗ**:		У оквиру рада имплементиран је кодер за пренос рендеринг коефицијената у ком се применом једне методе стеганографије коефицијенти утискују у аудио податке на процесору за дигиталну обраду сигнала фирме „Сирус Лоџик“.
Датум прихватања теме, **ДП**:		
Датум одбране, **ДО**:		05.09.2022.
Чланови комисије, **КО**:	Председник:	Доц. др Миодраг Ђукић
	Члан:	Проф. др Иван Каштелан
	Члан, ментор:	Доц. др Јелена Ковачевић

Потпис ментора

UNIVERSITY OF NOVI SAD **FACULTY OF TECHNICAL SCIENCES**
21000 NOVI SAD, Trg Dositeja Obradovića 6

KEY WORDS DOCUMENTATION

Accession number, **ANO**:			
Identification number, **INO**:			
Document type, **DT**:		Monographic publication	
Type of record, **TR**:		Textual printed material	
Contents code, **CC**:		Bachelor Thesis	
Author, **AU**:		Jovana Jezdimirović	
Mentor, **MN**:		PhD Jelena Kovačević	
Title, **TI**:		Implementation of rendering coefficient transmission coder based on embedding metadata in audio data on DSP	
Language of text, **LT**:		Serbian	
Language of abstract, **LA**:		Serbian	
Country of publication, **CP**:		Republic of Serbia	
Locality of publication, **LP**:		Vojvodina	
Publication year, **PY**:		2022	
Publisher, **PB**:		Author's reprint	
Publication place, **PP**:		Novi Sad, Dositeja Obradovica sq. 6	
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)		6/34/15/4/23/0/0	
Scientific field, **SF**:		Electrical Engineering	
Scientific discipline, **SD**:		Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, **S/KW**:		Digital signal processing, rendering, steganography	
UC			
Holding data, **HD**:		The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, **N**:			
Abstract, **AB**:		This paper covers the implementation of rendering coefficient transmission coder which uses an audio steganography method to embed rendering coefficients in audio data on Cirrus Logic digital signal processor.	
Accepted by the Scientific Board on, **ASB**:			
Defended on, **DE**:		05.09.2022.	
Defended Board, **DB**:	President:	PhD Miodrag Đukić	
	Member:	PhD Ivan Kaštelan	Menthor's sign
	Member, Mentor:	PhD Jelena Kovačević	

Захвалност

Захваљујем се својој менторки, доц. др Јелени Ковачевић и техничком ментору Ненаду Пекезу на стручној помоћи и подршци приликом израде овог рада.

Такође се захваљујем својој породици и пријатељима на сталној подршци коју су ми пружали током студирања.

САДРЖАЈ

Списак слика

Списак табела

Скраћенице

DSP – Дигитална обрада сигнала (Digital Signal Processing)

ADC – Аналогно-дигитални конвертор (Analog-to-Digital Converter)

DAC – Дигитално-аналогни конвертор (Digital-to-Analog Converter)

MSB – Бит највећег значаја (Most Significant Bit)

LSB – Бит најмањег значаја (Least Significant Bit)

WAVE – *Waveform* формат аудио датотеке (Waveform Audio File Format)

SBM – Супер бит мапирање (Super Bit Mapping)

LPF – Нископропусни филтар (Low-Pass Filter)

HPF – Високопропусни филтар (High-Pass Filter)

FIR – Коначан импулсни одзив (Finite Impulse Response)

CRC – Циклична провера редундансе (Cyclic Redundancy Check)

CLIDE – Интегрисано развојно окружење *Cirrus Logic* (Cirrus Logic Integrated Development Environment)

PCM – Импулсна кодна модулација (Pulse Code Modulation)

MIPS – Милион инструкција у секунди (Million Instructions Per Second)

1. Увод

Задатак овог рада је реализација кодера за пренос рендеринг коефицијената заснованог на утискивању метаподатака у аудио податке на *DSP*. Било је потребно осмислити протокол за пренос рендеринг коефицијената и имплементирати модул за кодовање тих информација на *DSP* платформи заснованој на чипу „Сирус Лоџика“ (енгл. *Cirrus Logic*). Имплементација је подразумевала креирање референтног модела у *C* програмском језику, а затим његово прилагођење циљној архитектури и реализацију у асемблерском језику. Примењена је методологија за развој *DSP* апликација која је развијена на Одсеку за рачунарску технику и рачунарске комуникације на Факултету техничких наука у Новом Саду.

Рендеринг коефицијенти представљају конфигурације за које ће се аудио сигнал рендеровати, тј. параметре на основу којих ће се вршити завршна обрада, која је у овом раду филтрирање. Задаје их корисник. У овом раду су то вредности појачања и граничне фреквенције филтра пропусника опсега. Рендеринг коефицијенти се у кодеру утискују у аудио сигнал над којим је потребно извршити обраду и на тај начин се као метаподаци заједно са аудио сигналом могу послати декодеру у ком се врши рендеринг. Коефицијенти се у аудио утискују на такав начин да људско ухо не може чути њихово присуство у аудио сигналу – примењена је аудио стеганографија.

Рад се састоји из 7 поглавља. У другом поглављу су дате теоријске основе о дигиталном звуку, субјективној перцепцији звука и дигиталној стеганографији. У трећем поглављу је представљен концепт решења. Четврто поглавље описује ток израде пројекта, као и главне структуре и функције реализованог модула и начине оптимизовања имплементираног кода. У петом поглављу су изнети резултати рада и

описани су начини верификације исправности реализованог модула. Шесто поглавље даје закључак рада, док седмо поглавље даје приказ литературе коришћене приликом израде рада.

2. Теоријске основе

У овом поглављу представљене су основе дигиталног аудија и субјективне перцепције звука. Када је реч о перцепцији звука, дотаћи ћемо се несавршености људског слушног система и начина на које та несавршеност може да се искористи у корист дигиталних аудио технологија. Тако ћемо доћи и до стеганографије, једне од техника које се ослањају на поменуте несавршености. Дат је увод у стеганографију и преглед стеганографских метода које су се користиле кроз историју. Затим су поменути и објашњени неки од типова стеганографије, а фокус је на аудио стеганографији. Поменуте су неке од основних метода аудио стеганографије, а међу њима се налази и метода на којој је фокус овог рада – метода кодовања бита најмањег значаја.

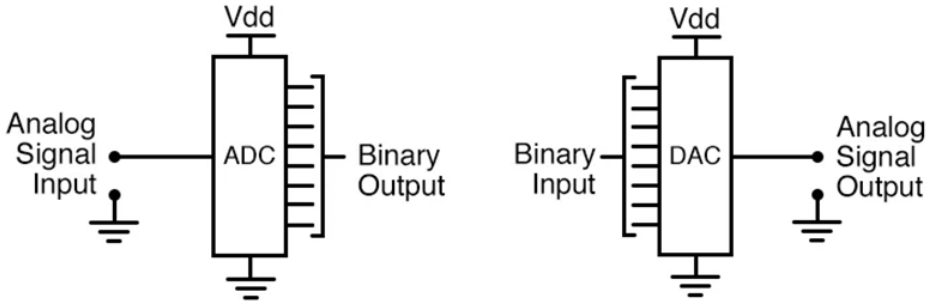
2.1 Дигитални аудио

Звук је по својој физичкој природи аналогни сигнал и да би се њиме могло манипулисати помоћу рачунара потребно је претворити га у дигитални облик. Процес дигитализације сигнала врши електронско коло које се назива аналогно-дигитални конвертор, или скраћено А/Д конвертор. А/Д конверзија обухвата дискретизацију по времену, дискретизацију по амплитуди и кодовање.

Дискретизација по времену се назива одабирање и представља процес очитавања сигнала у одређеним временским тренуцима. Фреквенција одабирања мора бити бар 2 пута већа од највеће фреквенције у сигналу да не би дошло до преклапања спектра (алијасинг), односно да би се реконструкција сигнала могла правилно извршити. Након одабирања ради се дискретизација по амплитуди, тзв. квантизација. Квантизација је поступак у ком се одабране вредности заокружују на најближу вредност из коначног

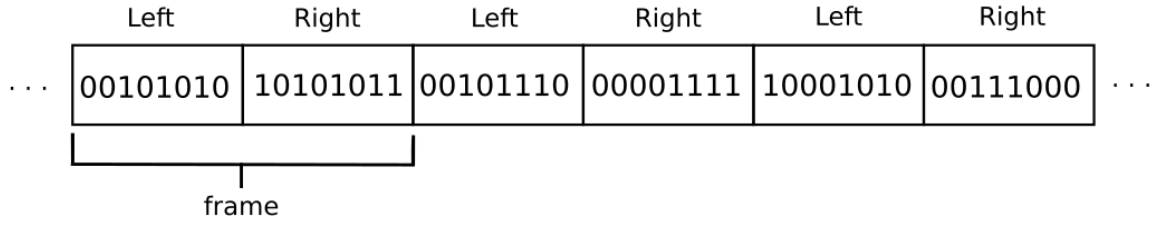
скупа вредности. На крају се ради кодовање, где се квантизовани одбирци представљају бинарним бројевима.

Ако желимо да репродукујемо дигитални звучни сигнал потребно је да га поново претворимо у аналогни облик. Процес претварања дигиталног сигнала у аналогни назива се Д/А конверзија, а коло које врши овај процес назива се Д/А конвертор.



Слика 2.1 Блок дијаграм А/Д и Д/А конвертора [1]

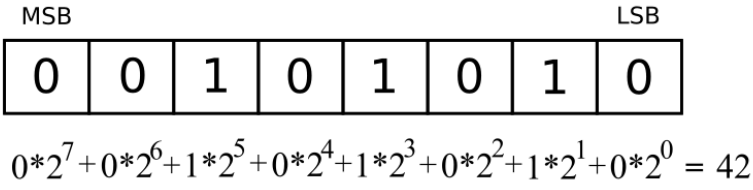
Постоји више типова аудио формата у којима се дигитални звук може чувати. Један од основних је *WAVE* (Waveform Audio File Format), познат и као *WAV* због .wav екстензије датотеке. Овај формат осим одбирака дигиталног сигнала садржи и заглавље у ком се налазе подаци о сигналу као што су фреквенција одабирања, величина одбирка, број канала и сл. Подаци у *WAV* формату су најчешће некомпримовани па је погодан за коришћење у аудио инжењерству јер се поред основне обраде не захтева никакво додатно енкодовање и декодовање. Најчешће се сусреће 16-битни *WAV* (величина одбирка је 16 бита) са фреквенцијом одабирања 44кHz. На слици 2.2 је приказан низ од неколико одбирака. Ради поједностављења илустрације приказан је 8-битни *WAV*.



Слика 2.2 Низ одбирака у 8-битном *WAVE* формату

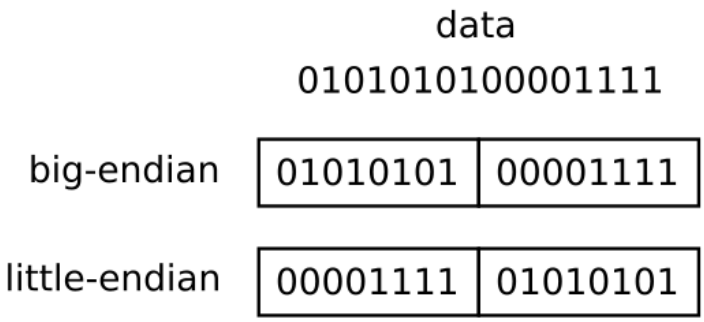
У питању је стерео сигнал (2 канала) и може се видети да су одбирци испреплетани - први одбирак левог и први одбирак десног канала представљају један оквир, други одбирак левог и други одбирак десног канала представљају други оквир итд. Сваки одбирак представља вредност амплитуде сигнала у одређеном временском тренутку. Са 8 бита могуће је приказати 2^8 различитих вредности, односно вредности из опсега од 0 до 255 ако су у питању неозначени бројеви или од -127 до 128 ако су у

питању означени бројеви. Код бинарне представе бројева сваки бит носи одређену тежину која зависи од његове позиције. Бит који има највећу тежину се најчешће налази на крајњој левој позицији и назива се *MSB* (Most Significant Bit), а бит који има најмању тежину налази се на крајњој десној позицији и назива се *LSB* (Least Significant Bit). Децимална вредност бинарног броја се рачуна тако што се саберу све тежине бита који имају вредност 1. Поступак рачунања децималне вредности осмобитног броја је приказан на слици 2.3.



Слика 2.3 Рачунање децималне вредности

Осам бита једног бинарног броја чини један бајт. Постоји 2 начина распоређивања података у меморији – *little-endian* и *big-endian*. Ако се подаци у меморији распоређују тако да је на нижој меморијској адреси нижи бајт податка онда се тај начин распоређивања назива *little-endian*, а ако се подаци распоређују тако да је на нижој адреси виши бајт податка онда се тај начин распоређивања назива *big-endian*.



Слика 2.4 *Little-endian* и *big-endian* начин распоређивања бајтова

2.1.1 Аудио рендеринг и рендеринг коефицијенти

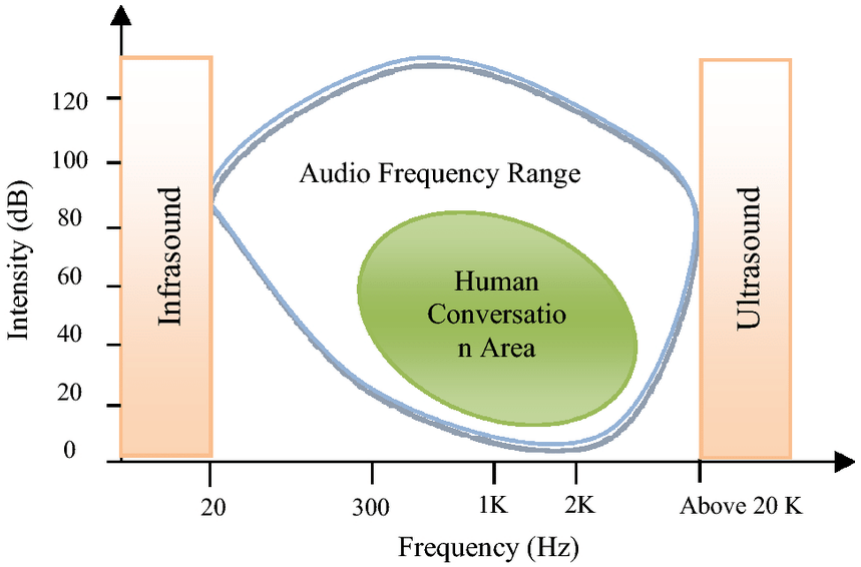
Појам рендеринг означава процес генерисања коначног дигиталног производа на основу одређеног типа улаза. Овај појам се најчешће сусреће у рачунарској графици, где се односи на генерисање дводимензионалних или тродимензионалних слика на основу претходно креираних модела применом одређених програма, али се такође користи и у аудио техници. Рендеринг аудио сигнала представља завршну обраду сигнала. Најчешће подразумева додавање ефеката као што су реверберација, промена

броја аудио канала, филтрирање и сл. Обрада се врши на основу рендеринг коефицијената који представљају конфигурације за које ће се аудио сигнал рендеровати. То могу бити *upmixer* или *downmixer* коефицијенти, коефицијенти за дигитално филтрирање или неки други параметри, у зависности од завршне обраде која се примењује.

2.2 Перцепција звука и аудио техника

Познавање људског чујног система игра значајну улогу у аудио инжењерству. Ово поглавље има за циљ да нас упозна са основама перцепције звука и покаже на које начине несавршености људског чула слуха могу бити искоришћене.

Фреквенцијски опсег чујности људског уха износи од 20 Hz до 20 kHz. Међутим, људи најбоље чују фреквенције у опсегу између 3000 Hz и 4000 Hz, што се може видети на слици испод (Слика 2.5).



Слика 2.5 Опсег чујности

Праг чујности представља најнижи звучни притисак (интензитет звука) који људско ухо може да региструје и зависи од фреквенције. Са слике 2.5 се може уочити и то да је праг чујности на веома ниским и високим фреквенцијама виши, тј. да је осетљивост људског уха на тим фреквенцијама мања. Ове карактеристике примењују се у техникама као што је „уобличавање шума“, која мења спектар квантизационог шума, односно обликује га тако да већина шума буде сконцентрисана на високим фреквенцијама, где га људско ухо скоро и не опажа. Ова техника има за циљ да повећа однос сигнал-шум датог сигнала и користи се за повећавање резолуције А/Д и Д/А конвертора. Још један од познатих примера где се уобличавање шума користи је и

Сонијев *SBM* (Super Bit Mapping) - технологија која омогућава конвертовање 20-битног сигнала у 16-битни скоро без губитка квалитета звука [3].

Интензитет звука које људско ухо може да детектује мери се у децибелима (dB). Динамички опсег чујности представља разлику између прага бола и прага чујности и износи 120 dB. Међутим, иако људски чујни систем има велики динамички опсег, његов диференцијални опсег је прилично мали, што значи да јачи звукови могу маскирати слабије [4]. Маскирање је појава која се примењује код метода компресије звучних сигнала као што је *MP3*, а примењује се и код неких метода скривања информација унутар аудио датотека. Важно је напоменути и да је људско ухо неосетљиво на промену фазе, што је још једна од особина које се могу искористити код неких техника компресије сигнала и скривања тајних информација.

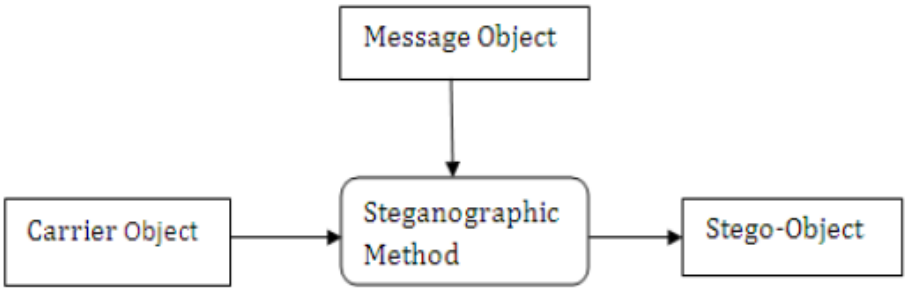
Ухо је такође неосетљиво на измене бита најмањег значаја (енгл. Least Significant Bit, LSB). Знамо да се код дигиталних аудио формата као што је *WAV* одбирци сигнала изражавају у бинарној форми и представљају амплитуду сигнала. Ако изменимо *LSB* бит једног одбирка, његова децимална вредност ће се увећати или умањити за 1, што заиста није значајна измена. Измена *LSB* бита уноси адитивни Гаусов шум у аудио сигнал, али је тај шум довољно мали да људско ухо не може да га чује. Ипак, људско ухо је јако осетљиво на шум па је број *LSB* бита које је могуће изменити а да не дође до деградације аудио сигнала ограничен.

2.3 Стеганографија

Стеганографија је техника скривања порука унутар других порука. Термин стеганографија се први пут јавио 440. године п.н.е. а настао је од грчких речи *steganos* (гр. στεγανός), што значи покривен, и *grafia* (гр. γραφία), што значи писање. У раним 1800-тим је обележен као архаични назив за криптографију, али је 1980-тих година враћен у живот као реч која означава врсту дигиталне криптографије [5]. Оно због чега се стеганографија издваја као посебан облик криптографије је то што она за циљ има да сакрије не само садржај поруке која се шаље, већ и само постојање те поруке.

Основни концепт и сврха стеганографије могу се објаснити на примеру „проблема затвореника“ који је дао Густав Симонс (енгл. Gustavus Simmons) 1983. године: Два затвореника покушавају да нађу начин како да побегну из затвора, али проблем им представља чувар који надгледа њихову комуникацију. Ако чувар сазна да постоји неки вид тајне комуникације послаће их у самицу и онда неће моћи да комуницирају уопште. Због тога они морају да смисле како да комуницирају на такав начин да чувар не примети да се скривене поруке размењују. Идеја је да се тајна порука сакрије унутар

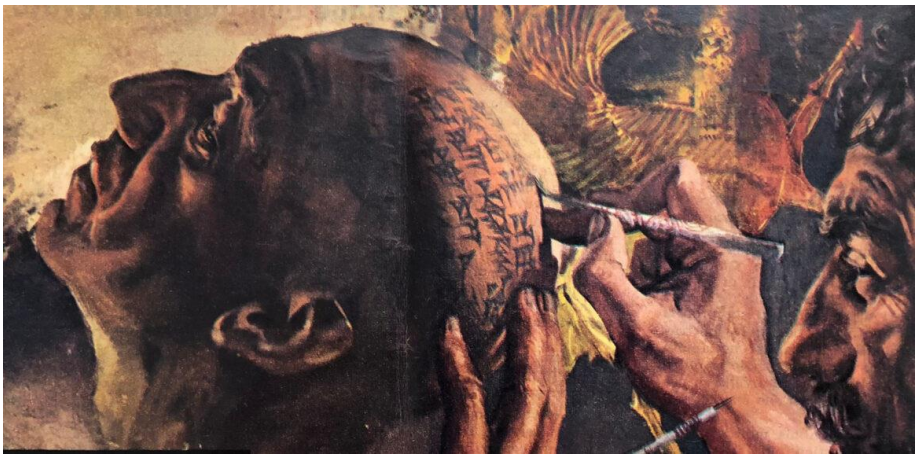
неке друге, наизглед безазлене, поруке [6]. Симонсов рад је био први рад који је објављен на тему дигиталне стеганографије и са њим је научни приступ изучавања стеганографије практично започет.



Слика 2.6 Основни блок дијаграм стеганографије

2.3.1 Стеганографија кроз историју

Први примери примене стеганографије датирају још из доба старе Грчке. Један од примера је коришћење дрвених плоча преливених воском. Тајна порука би се написала на дрвену плочу, а затим би се плоча прелила воском и порука би остала скривена док се восак не уклони са ње. Други пример је тетовирање главе гласника. Гласник би обријао главу како би се на њу истетовирала порука, а онда би се сачекало да коса порасте како би гласник неприметно пренео скривену поруку. Када би стигао на одредиште тамо би опет обријао главу како би се порука прочитала. На сличан начин су и древни Кинези преносили информације користећи своје гласнике као преносни медијум. Наиме, порука би се написала на танак комад свиле, који би био стављен у воштану куглу коју би гласник прогутао и тако пренео поруку. [8].



Слика 2.7 Тетовирање главе гласника [9]

У новијој историји постоје многобројни примери из Другог светског рата. Немачки шпијуни су користили такозване нулте шифре, метод текстуалне стеганографије у ком се текст тајне поруке скривао унутар друге, наизглед обичне, текстуалне поруке. Такође се користило и невидљиво мастило, а измишљене су и „микротачке“ (енгл. microdots) – фотографије са тајном поруком умањене на величину пречника 1 милиметар, односно величину тачке коју откуца писаћа машина [8].

Са развојем информационо-комуникационих технологија омогућен је пренос података путем различитих медијума, па је самим тим и стеганографија добила више врста носиоца поруке. Тако се данас не користе восак, дрвене плоче или тела људи, већ се примењују методе дигиталне стеганографије којима се поруке утискују у дигиталне фотографије, аудио датотеке, видео и сл. Мултимедијалне датотеке су врло погодне за примену стеганографије због својих величина јер чине измене извршене над њима (скоро) неприметним.

Можемо приметити да се кроз историју стеганографија најчешће користила у сврхе шпијунаже. Међутим, данас ова техника има много ширу примену. Користи се за заштиту ауторских права (дигитални водени жиг), заштиту података од измене, пренос података, у медицини за чување поверљивих података пацијената и многе друге сврхе.

2.3.2 Типови дигиталне стеганографије

Скоро сви дигитални формати могу бити коришћени за стеганографију, али најпогоднији су они који имају висок степен редундантности, односно они који садрже велики број бита које је могуће изменити, а да притом измена не буде лако уочљива [10]. На основу тога издваја се 5 основних типова стеганографије, а то су: стеганографија у тексту, стеганографија у дигиталним фотографијама, стеганографија у видео форматима, стеганографија у аудио форматима и стеганографија у мрежним протоколима.

Стеганографија у тексту подразумева скривање информација унутар текстуалних датотека. Користе се 3 врсте метода: метода заснована на формату, случајна и статистичка метода и лингвистичка метода [11].

Стеганографија у аудио форматима представља утискивање информација у дигитални звук. То се постиже вршењем малих измена над бинарном секвенцом аудио датотеке. О овој врсти дигиталне стеганографије биће више речи у наредном поглављу јер је на њој фокус овог рада.

Стеганографија у дигиталним фотографијама је најпопуларнији тип стеганографије јер дигиталне фотографије садрже велики број редундантних бита.

Подразумева мењање интензитета пиксела са циљем утискивања скривене поруке у слику. Најчешће се примењује на BMP, PNG, JPEG и GIF форматима.

Стеганографија у видео форматима користи MP4, AVI, MPEG и друге типове видео формата као носиоце тајне поруке. Дигитални видео је аудио-визуелни формат, па тако све технике које се примењују код стеганографије у дигиталним фотографијама и аудио форматима најчешће могу бити примењене и овде [12].

Стеганографија у мрежним протоколима подразумева скривање информација унутар мрежних протокола као што су TCP, UDP, ICMP, IP итд. На пример, порука се може сакрити у заглавље TCP/IP пакета у неком пољу које је опционо [11].

2.3.3 Аудио стеганографија

Као што је у претходном поглављу већ наведено, аудио стеганографија подразумева скривање порука унутар аудио датотека. Аудио стеганографија се ослања на несавршености људског чула слуха. Овај тип стеганографије је доста захтеван због тога што су фреквенцијски (1:1000) и динамички (120 dB) опсег чујности велики и људско ухо је јако осетљиво на промене. Може се упоредити са чулом вида чији је фреквенцијски опсег у размери 1:2, а динамички опсег је 90 dB. Ипак, као што је у поглављу 2.2 показано, постоје карактеристике слушног система које се могу искористити у сврхе примене стеганографије. Предности аудио стеганографије су то што аудио датотеке имају висок степен редундантности и великог су капацитета па је у њих могуће утиснути велике количине информација.

2.3.3.1Технике аудио стеганографије

Постоји велики број различитих техника аудио стеганографије, али у даљем тексту је дат преглед само неких од основних.

Кодовање паритета (енгл. parity coding) – дели сигнал на више одвојених група одбирака и утискује бит тајне поруке у бит парности групе одбирака. Ако се бит парности групе не подудара са битом тајне поруке који је потребно утиснути онда се мења вредност *LSB* бита неког од одбирака из групе. Предност ове методе је то што је рачунски једноставна, а мана је мала отпорност на измене које могу настати приликом обраде сигнала [13].

Кодовање фазе (енгл. phase coding) – заснива се на чињеници да је људско ухо неосетљиво на фазу. Ради тако што мења фазу почетног аудио сегмента са референтном фазом која представља податке. Фазе узастопних сегмената се прилагођавају тако да се задржи релативна фаза између сегмената. У поређењу са осталим методама има најбољи однос сигнал-шум [13].

Метод проширеног спектра (енгл. spread spectrum coding) – користи чињеницу да је мале промене теже спазити на већим енергетским нивоима. Покушава да скривене податке рашири по спектру што је више могуће. То омогућава пријем сигнала чак и ако дође до интерференције на неким фреквенцијама. Мана ове методе је то што се може десити да унесе шум у аудио сигнал [13].

Кодовање бита најмањег значаја (енгл. LSB coding) – најједноставнији начин за утискивање информација у дигитални аудио. Сваки *LSB* се мења једним битом бинарне поруке. Применом ове методе могуће је утиснути велике количине података. Мана је мала отпорност на измене које могу настати приликом обраде сигнала. Постоје различите варијанте ове методе, а једна од њих је и примена логичке операције ексклузивно или над битима најмањег значаја, која је искоришћена у овом раду.

2.3.3.2 Примена логичке операције ексклузивно или над битима најмањег значаја (XORing LSBs)

Ова метода се заснива на примени операције ексклузивно или над битима најмањег значаја. У зависности од резултата операције ексклузивно или над *LSB* битима и у зависности од вредности бита који желимо да утиснемо, вредност *LSB* бита се мења или остаје иста. Процес утискивања података описује табела испод (Табела 2.1).

LSB	Bit next to LSB	XOR	Action if message bit is 0	Action if message bit is 1
0	0	0	No Change	Flip LSB
0	1	1	Flip LSB	No Change
1	0	1	Flip LSB	No Change
1	1	0	No Change	Flip LSB

Табела 2.1 Процес утискивања података [14]

Довољно је користити 2 *LSB* бита, али ако желимо да повећамо јачину шифровања можемо користити чак до 16 *LSB* бита [14]. Основна предност ове методе је то што је једноставна за имплементацију јер не захтева комплексне математичке прорачуне.

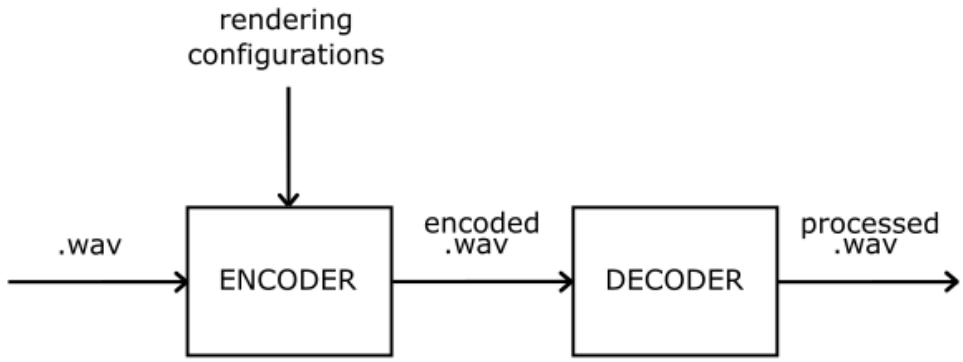
3. Концепт решења

Тема овог рада је реализација кодера за пренос рендеринг коефицијената.
Циљ је да се коефицијенти који се користе за рендеринг (завршну обраду) аудио сигнала пошаљу блоку за обраду заједно са тим сигналом, тачније унутар њега. Да би се то постигло примењена је једна од стеганографских метода. Коефицијенти су утиснути у аудио сигнал на такав начин да се никакве значајне промене у сигналу не примећују. Осмишљен је протокол за пренос података који се утискују у сигнал.

3.1 Кодер - декодер систем

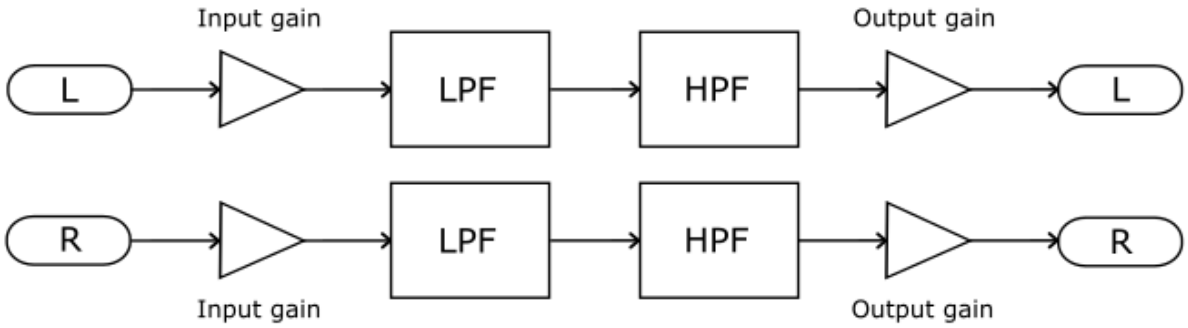
Утискивање података се врши у кодерском модулу, а за потребе тестирања исправности рада кодера реализован је и декодерски модул у ком се врши реконструкција утиснутих података и обрада датог сигнала.

Као улаз у кодер имамо .wav датотеку. У њу се у овом модулу подаци потребни за обраду аудио сигнала утискују применом аудио стеганографије. На излазу се добија датотека са изменама довољно малим да људско ухо не примети разлику између ње и улазне датотеке. Излаз из кодера представља улаз у декодер. У њему се врши реконструкција података који су у кодеру утиснути у датотеку. Такође се врши и обрада улазног сигнала на основу реконструисаних података.



Слика 3.1 Енкодер и декодер

Обрада коју декодерски модул врши над сигналом је иста за оба аудио канала и подразумева филтрирање филтром пропусником опсега који је добијен везивањем на ред нископропусног (енгл. Low-Pass Filter, LPF) и високопропусног (енгл. High-Pass Filter, HPF) FIR (Finite Impulse Response) филтра другог реда. Детаљан приказ система за обраду је дат на слици 3.2.



Слика 3.2 Блок дијаграм система за обраду

Параметри које корисник може да мења, односно наши рендеринг коефицијенти, су јачина улазног појачања, јачина излазног појачања, гранична фреквенција нископропусног филтра и гранична фреквенција високопропусног филтра. За пренос ових параметара осмишљен је посебан протокол, о ком ће више речи бити у наредном поглављу.

3.2 Опис протокола и формат пакета

Пакет је низ битских поља енкодованих у *little-endian* формату. Сваки пакет се састоји од четири поља. То су:

- payload ID - поље које садржи идентификациони број пакета,
- payload length - поље које садржи величину пакета,
- variable data - поље које садржи податке, тј. рендеринг коефицијенте и
- CRC - поље које садржи код за контролу тачности

Поља *payload ID, payload length* и *CRC* за сваку врсту пакета имају исти број бајтова, док се величина поља *variable data* мења у зависности од врсте пакета.

FIELD:	PAYLOAD ID	PAYLOAD LENGTH	VARIABLE DATA	CRC
LENGTH (Bytes):	1	1	{2, 4}	1

Слика 3.3 Формат пакета

У реализованом протоколу постоји 3 врсте пакета: *payload 0, payload 1* и *payload 2*.

Пакет са идентификационим бројем 0 (*payload 0*) у свом *variable data* пољу садржи 32-битну синхронизациону реч. Она служи за означавање почетка низа пакета. Ако декодер не пронађе синхронизациону реч цео низ пакета се одбацује и реконструкција података које су ти пакети садржали неће моћи да се изврши.

Пакет *payload 1* садржи податке о граничним фреквенцијама нископропусног и високопропусног филтра и величине је 2 бајта – 1 бајт за код који представља одређену фреквенцију LP филтра и 1 бајт за код који представља одређену фреквенцију HP филтра. Граничне фреквенције које можемо задати за LP филтар су 12 kHz, 14 kHz, 16 kHz, 18 kHz и 20 kHz, а за HP филтар су 2 kHz, 4 kHz, 6 kHz, 8 kHz и 10 kHz и кодују се редом бројевима од 0 до 4. Дакле, ако су граничне фреквенције пропусника опсега вредности 4 kHz и 18 kHz, онда ће се у *variable data* пољу пакета *payload 1* налазити вредности 1 и 3.

HPF cutoff frequency [kHz]	LPF cutoff frequency [kHz]	Code
2	12	0
4	14	1
6	16	2
8	18	3
10	20	4

Табела 3.1 Граничне фреквенције филтра

Пакет *payload 2* садржи вредности улазног и излазног појачања. Величина *variable data* поља овог пакета је 4 бајта – 2 бајта за улазно и 2 бајта за излазно појачање.

PAYLOAD 0

PAYLOAD 0	PAYLOAD LENGTH	SYNC WORD	CRC
1	1	4	1

PAYLOAD 1

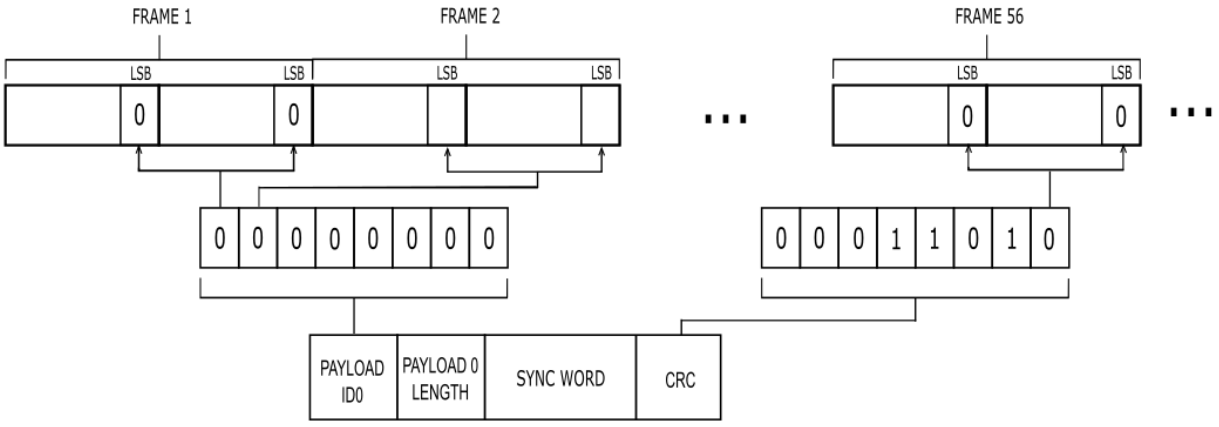
PAYLOAD 1	PAYLOAD LENGTH	LP CUTOFF	HP CUTOFF	CRC
1	1	2		1

PAYLOAD 2

PAYLOAD 2	PAYLOAD LENGTH	INPUT GAIN	OUTPUT GAIN	CRC
1	1	4		1

Слика 3.4 Врсте пакета

Пакети се шаљу редом *payload 0*, *payload 1*, *payload 2*. У сваки оквир аудио тока се утискује по један бит пакета. Тако је за утискивање пакета *payload 0* потребно 56 оквира аудио тока зато што се у сваки оквир утискује по 1 бит, а пакет *payload 0* се састоји од 7 бајтова (7 * 8 бита). За утискивање пакета *payload 1* потребно је 40 оквира, а за утискивање пакета *payload 2* потребно је 56 оквира. Дакле, на свака 152 оквира у аудио ток ће се утиснути цео низ пакета.

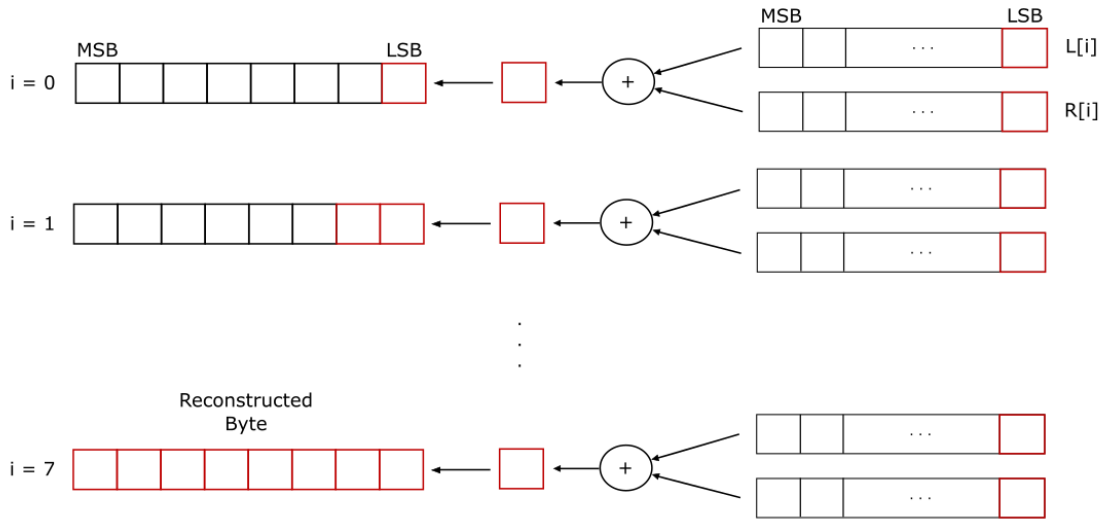


Слика 3.5 Утискивање једног пакета у аудио ток

Због чега је за утискивање једног бита података потребан 1 оквир аудио тока, а не само један одбирак, биће објашњено у наредном поглављу у ком се описује начин утискивања података.

3.3 Утискивање бита

Као што је раније споменуто, за утискивање података у аудио ток у овом раду је искоришћена стеганографска метода кодовања бита најмањег значаја, конкретно примена логичке операције ексклузивно или над битима најмањег значаја. За сваки бит података који желимо да утиснемо потребан је један аудио оквир, односно по један одбирак левог и десног канала, тачније њихови *LSB* бити.



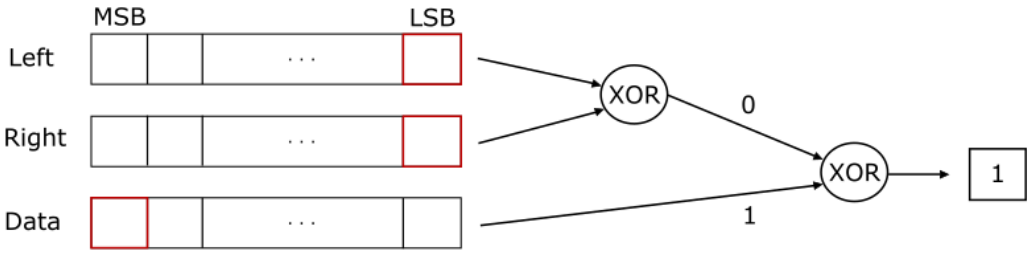
Слика 3.6 Реконструкција једног бајта података

Како би се лакше разумео метод утискивања података у аудио датотеку боље је прво објаснити на који начин се скривени подаци реконструишу. Подаци се реконструишу бајт по бајт, а реконструисани бајтови се креирају од битова који се добијају као резултат сабирања LSB бита одбирака левог и десног канала. То се ради тако што се у свакој итерацији бит који се добије тим сабирањем поставља на најнижу позицију у бајту, а затим се све помера за једно место у лево. Поступак се понавља 8 пута, док се не реконструише цео бајт. Дакле, ако вредност неког бита у бајту желимо да поставимо на 1 онда на кодерској страни морамо да обезбедимо да LSB бити одбирка левог и десног канала имају различите вредности, а ако желимо нулу онда LSB бити левог и десног канала морају бити исти.

Као што се из претходно описаног поступка може закључити, оно што ће нас занимати при утискивању података у аудио сигнал су вредности битова бајта који утискујемо и однос LSB бита левог и десног канала. На основу њих се на кодерској

страни рачуна грешка утискивања која нам говори да ли је однос бита одговарајући или је потребно променити га. Овде можемо разликовати 4 случаја:

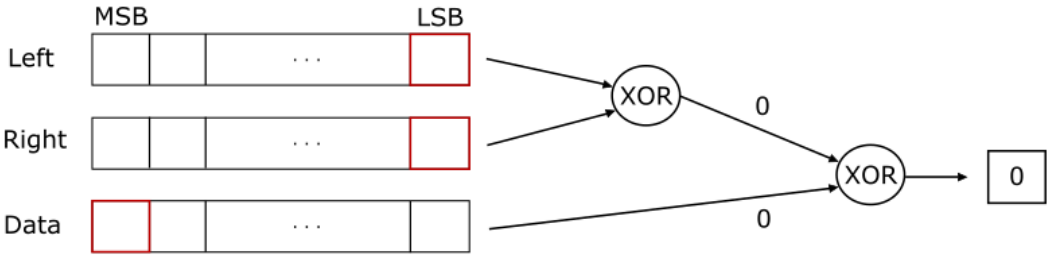
1) MSB бајта који желимо да утиснемо је 1, а LSB одбирка левог и десног канала су једнаки



Слика 3.7 Рачунање грешке утискивања, први случај

У овом случају сматрамо да однос LSB бита није одговарајући јер су они једнаки, па бисмо при реконструкцији њиховим сабирањем добили резултат 0, а бит који желимо да утиснемо је 1. Грешка утискивања ће бити једнака јединици, што значи да се однос бита мора мењати.

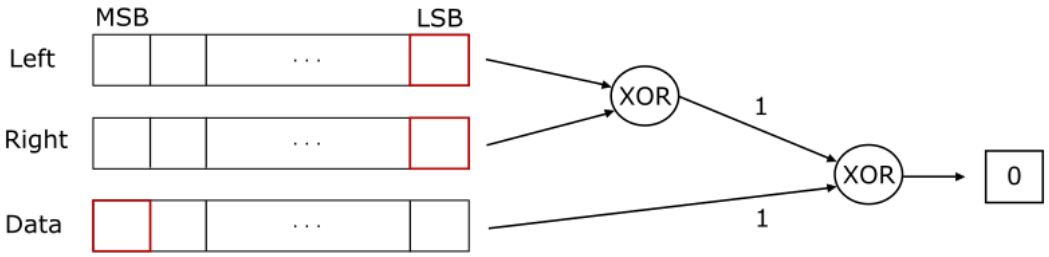
2) MSB бајта који желимо да утиснемо је 0, а LSB одбирака левог и десног канала су једнаки



Слика 3.8 Рачунање грешке утискивања, други случај

Ово је случај у ком је однос бита одговарајући. LSB бити су једнаки, што сабирањем даје резултат 0, а бит који желимо да утиснемо јесте 0. Грешка утискивања је 0.

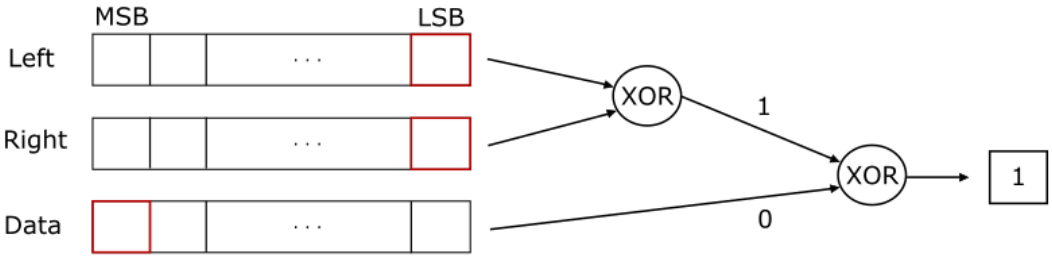
3) MSB бајта који желимо да утиснемо је 1, а LSB одбирака левог и десног канала нису једнаки



Слика 3.9 Рачунање грешке утискивања, трећи случај

Ни у овом случају није потребно мењати однос LSB бита. Различити бити у збиру дају резултат 1, а бит који утискујемо је 1. Грешка је 0.

4) MSB бајта који желимо да утиснемо је 0, а LSB одбирака левог и десног канала нису једнаки



Слика 3.10 Рачунање грешке утискивања, четврти случај

У последњем случају је потребно мењати однос бита јер желимо да утиснемо нулу, а њихов однос у збиру даје 1. Грешка утискивања је 1.

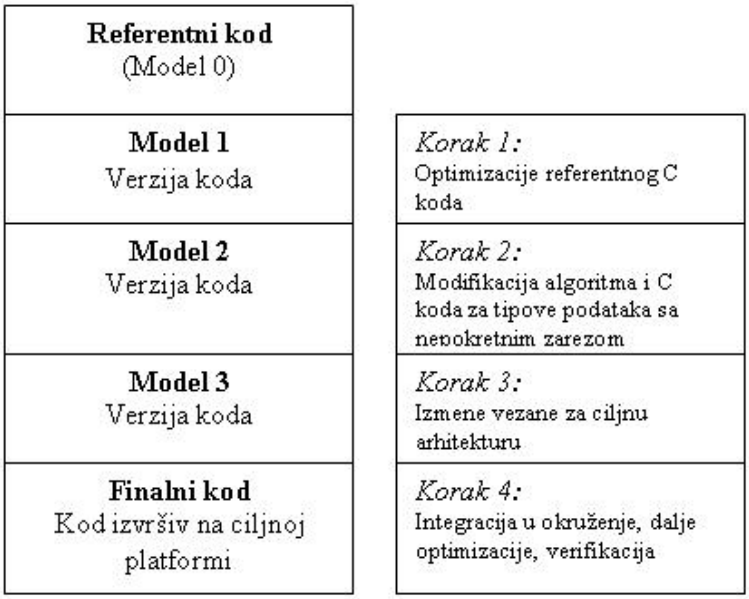
На основу грешке утискивања се одређује индекс из табеле претраге која садржи пар вредности које могу бити 0, 1 или -1. Те вредности се додају на одбирке левог и десног канала. На парним индексима се налазе парови вредности које када се додају на канале не утичу на однос *LSB* бита одбирака, а а на непарним индексима се налазе парови који мењају тај однос.

4. Програмско решење

Ово поглавље садржи опис фаза развоја DSP апликације кодера за пренос рендеринг коефицијената и процес имплементације и оптимизације решења у *C* и асемблерском програмском језику.

4.1 Методологија развоја DSP апликација

На дигиталним сигнал процесорима се реализује широк спектар апликација: аудио и видео кодери, алгоритми заштите говора, модеми, елиминација шума, препознавање говора итд. Алгоритми се најчешће развијају на процесорима опште намене у неком од виших програмских језика (*C* или *C++*) или у форми математичког модела. На тај начин се добија референтни код алгоритма, који није зависан од физичке платформе и представља основу за развој апликација за *DSP* процесоре. Да би тако развијени алгоритам могао да се преведе за циљну платформу потребно је да се итеративним поступком прилагоди. На Одсеку за рачунарску технику и рачунарске комуникације на Факултету техничких наука у Новом Саду је развијена методологија за развој софтвера за дигиталне сигнал процесоре, која прописује кораке који олакшавају и убрзавају развој и одржавање кода. Први корак је анализа улазних испитних вектора, анализа задатог алгоритма и, уколико је потребно, формирање референтног *C* кода. Резултат овог корака јесте референтни код, односно Модел 0 и скуп излазних референтних вектора који служе за проверу исправности сваке наредне фазе. Следећи корак, Модел 1, уводи функционалне оптимизације у *C* код. Модел 2 подразумева прилагођење кода аритметици циљне архитектуре, док Модел 3 представља потпуно преводив *C* код за наменску платформу. [15]



Слика 4.1 Ток имплементације софтвера на дигиталним сигнал процесорима са аритметиком у непокретном зарезу [15]

Код развоја кодерске апликације која је задатак овог рада примењена је описана методологија, са једном изменом: израда модела 3 је изостављена јер није било потребе за креирањем самосталног пројекта у *C* језику, већ је он имплементиран директно у асемблерском језику. Детаљан опис реализације сваке од фаза развоја дат је у наредним потпоглављима.

4.2 Фаза 1

У овој фази је у *C* програмском језику имплементиран алгоритам за утискивање података у аудио ток коришћењем стеганографске методе која подразумева примену логичке операције ексклузивно или над битима најмањег значаја. Развојно окружење које је коришћено је *Microsoft Visual Studio 2022*. Ово окружење омогућава контролисано извршавање програма, приказ меморије и многе друге опције које олакшавају развој апликације. У даљем тексту следи опис основних програмских структура и функција које су коришћене за реализацију поменутог алгоритма.

4.2.1 Структура *Payload1_t*

Ова структура представља *variable data* поље пакета *payload 1*. Садржи поља *nLPCutoffFreq* и *nHPCutoffFreq*, чије вредности представљају граничне фреквенције филтра пропусника опсега које корисник задаје.

4.2.2 Структура *Payload2_t*

Ова структура представља *variable data* поље пакета *payload 2*. Садржи поља *nInputGain* и *nOutputGain*, чије вредности представљају улазно и излазно појачање.

4.2.3 Структура *Packet*

Ова структура садржи поља *mCapacity, p_mData* и *mBytePosition* која описују један пакет осмишљеног протокола. Поље *mCapacity* садржи максималну величину пакета, поље *p_mData* је показивач на низ који садржи податке једног пакета, а поље *mBytePosition* садржи информацију о томе који бајт пакета је на реду за утискивање.

4.2.4 Структура *AudioAdj*

Ова структура садржи поља *lAdj* и *rAdj*, која представљају коефицијенте који се додају на одбирке левог и десног канала са циљем добијања одговарајућег односа њихових LSB бита.

4.2.5 Структура *Burier*

Ова структура садржи поља *adjLookup* и *ditherSource*. Поље *adjLookup* представља показивач на низ структура *AudioAdj*, који представља табелу претраге, а поље *ditherSource* представља псеудо случајни број на основу ког се рачуна индекс табеле претраге на коју *adjLookup* показује.

4.2.6 Структура *Bitstack*

Ова структура садржи поља *mPayload1* и *mPayload2* која представљају *variable data* поља пакета *payload 1* и *payload 2* и садрже рендеринг коефицијенте.

4.2.7 Структура *BitstackWriter*

Ова структура садржи поља *writePacket*, *bitstack*, *burier*, *pendingBits* и *pendingData*. Поља *writePacket*, *bitstack* и *burier* представљају структуре *Packet*, *Bitstack* и и *Burier*, које су објашњене у претходним поглављима. Поље pendingBits садржи информацију о томе колико бита података треба предати функцији која врши утискивање, а *pendingData* садржи податке који су на реду за утискивање.

4.2.8 Функција *main*

У главној *main* функцији се врши неколико једноставних ствари:

- учитавање *.wav* датотеке која садржи аудио сигнал у који је потребно утиснути рендеринг коефицијенте,

- учитавање рендеринг коефицијената које корисник задаје путем командне линије и
- позивање функције *BitstackWriter_write()* у којој се врши утискивање рендеринг коефицијената.

4.2.9 Функција *BitstackWriter_write*

Ова функција узима редом бајтове пакета и предаје их функцији *Burier_buryBits* у којој се одвија основна логика утискивања података. Такође проверава да ли су сви бајтови једног пакета послати, и ако јесу онда позива функцију *BitstackWriter_nextWritePacket()* у којој се поставља наредни пакет за слање.

4.2.10 Функција *BitstackWriter_nextWritePacket*

Ова функција се позива унутар функције *BitstackWriter_write()* и, као што је већ напоменуто, служи за постављање наредног пакета за слање. Проверава који је пакет последњи послат и на основу тога позива одговарајућу функцију за подешавање следећег пакета.

4.2.11 Функција *Burier_buryBits*

У овој функцији је имплементирана логика стеганографске методе засноване на измени бита најмањег значаја, конкретно методе примене операције ексклузивно или над битима најмањег значаја која је представљена у поглављу 2.3.3.2 у оквиру теоријских основа, а детаљније објашњена у поглављу 3.3 у оквиру концепта решења.

4.2.12 Функција *Packet_setPayload0*

У овој функцији се подешавају поља пакета payload 0, који треба да садржи синхронизациону реч.

4.2.13 Функција *Packet_setPayload1*

У овој функцији се подешавају поља пакета payload 1, који треба да садржи граничне фреквенције филтра пропусника опсега.

4.2.14 Функција *Packet_setPayload2*

У овој функцији се подешавају поља пакета payload 2, који треба да садржи јачину улазног и излазног појачања.

4.3 Фаза 2

Након успешно завршене прве фазе и креираног референтног модела (модел 0) прешло се на реализацију модела 1, у ком су извршене функционалне оптимизације кода имплементираног у референтном моделу. Ове оптимизације подразумевају измене у референтном коду у складу са хардверским проширењима процесора: броју регистара, величини меморије, величини стека, раду адресног генератора и хардверским петљама. Ове измене се своде на организацију кода и података, тако да након превођења утрошак процесорских ресурса буде оптималан и у складу са захтевима [15]. У даљем тексту су наведене оптимизације које су извршене.

Приликом превођења *C* кода, локалне променљиве се смештају у регистре процесора или на локални стек, а глобалне променљиве се смештају у радну меморију. С обзиром на то да циљна платформа има ограничен број регистара и малу величину стека, подаци су организовани тако да су, где год је то било прихватљиво, уместо локалних променљивих коришћене глобалне променљиве.

Осим тога, за смештање података на стек потребно је више инструкција него за смештање у програмску меморију, па је то посебно избегавано. Где год је то било могуће, смањен је број аргумената функције јер ако број аргумената функције пређе број слободних регистара онда се параметри смештају на стек. То се посебно односило на функције обраде, које се у току извршавања програма позивају велики број пута.

Још једна од оптимизација која је примењена, а која највише утиче на брзину извршавања кода, је измена начина приступања подацима. Операције приступа су прилагођене раду адресног генератора, што значи да је приступ елементима низа коришћењем индексирања замењен приступом преко показивача на елемент.

Оптимизоване су и програмске петље. Уочени су делови кода за које није било потребе да се налазе унутар петљи и тако беспотребно повећавају број извршених инструкција, а затим су измештени ван њих.

4.4 Фаза 3

У овој фази реализован је модел 2. Код је прилагођен аритметици циљне архитектуре. То је подразумевало замену свих типова података типом int32_t који представља најмању адресиву јединицу платформе, односно један бајт на тој платформи. Циљна платформа не подржава операцију дељења, па су сви изрази у којима је коришћена ова операција, а чији је резултат била константна вредност, унапред израчунати и замењени константама. На платформи није подржан ни тернарни

оператор, па је уместо њега коришћен *if-else* услов. *Switch-case* наредбе су такође замењене *if-else* условима.

4.5 Фаза 4

Четврта фаза је подразумевала писање асемблерског кода на основу *C* кода који је прилагођен архитектури циљне платформе. Коришћено је *CLIDE (Cirrus Logic Integrated Development Environment)* интегрисано развојно окружење које је развијено од стране *Cirrus Logic* компаније, а засновано је на *Eclipse* платформи. Ово окружење нуди велики број алата који олакшавају развој програмског кода. Омогућава контролисано извршавање програма, приказ меморије и приказ стања регистара. Такође омогућава превођење програмског кода и пружа подршку за извршавање *DSP* апликације у симулатору, као и на циљној платформи. Приликом писања асемблерског кода еквивалентног референтном *C* коду, вршена је и оптимизација која је подразумевала коришћење могућности паралелног извршавања инструкција и коришћење хардверских петљи где је то било могуће.

4.6 Фаза 5

Након завршене четврте фазе и успешно реализованог кодерског модула у асемблерском језику, уследила је интеграција са радним окружењем и оперативним системом циљне платформе.

5. Резултати

У овом поглављу су представљени методологија и резултати тестирања имплементираног решења. Такође су представљени резултати профилисања кода, тј. дате су информације о утрошеним ресурсима. Тестирање се, као и развој апликације, одвијало по фазама. Прво је извршено испитивање исправности референтног модела, а затим су на основу излазних вектора референтног модела испитани сви остали модели. Рађени су слушни, спектрални и бит-идентични тестови.

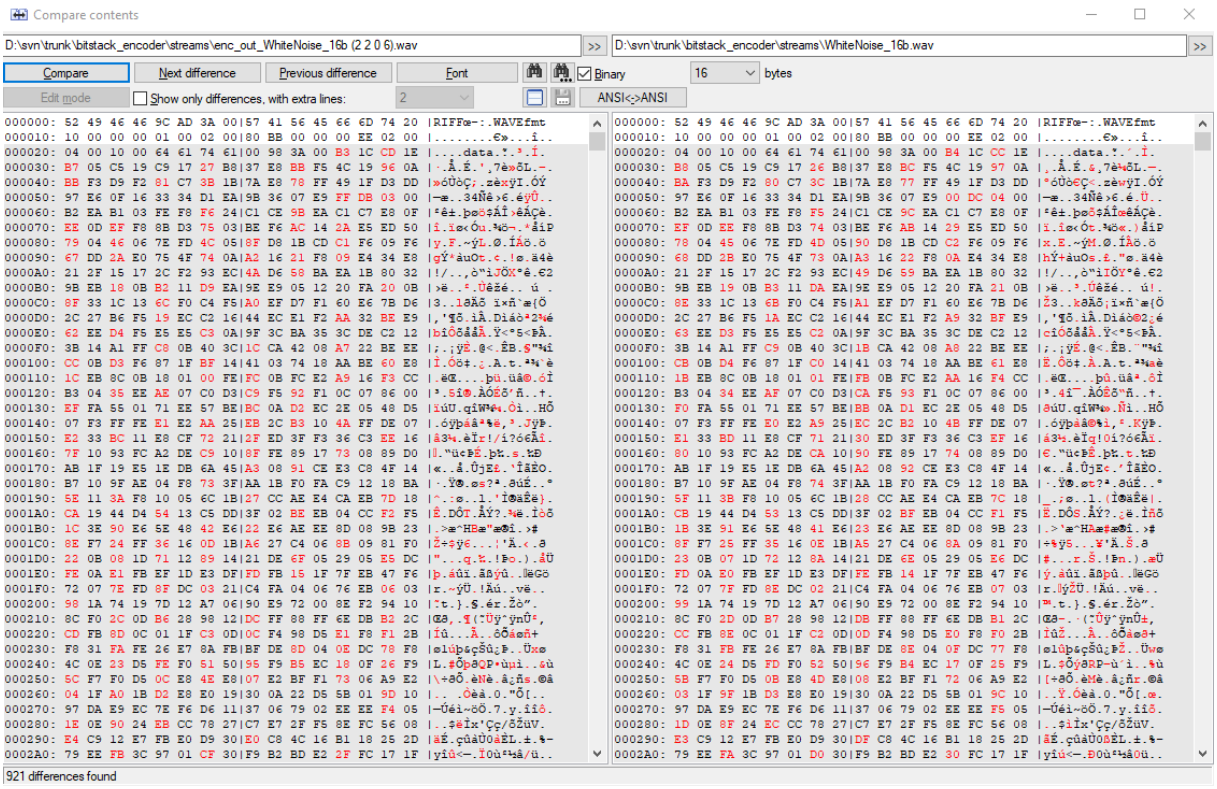
5.1 Испитивање референтног модела

Након што је референтни модел кодера за пренос рендеринг коефицијената имплементиран у *Visual Studio* развојном окружењу у *C* програмском језику, приступило се испитивању његове исправности. Исправно функционисање овог модула подразумева две ствари:

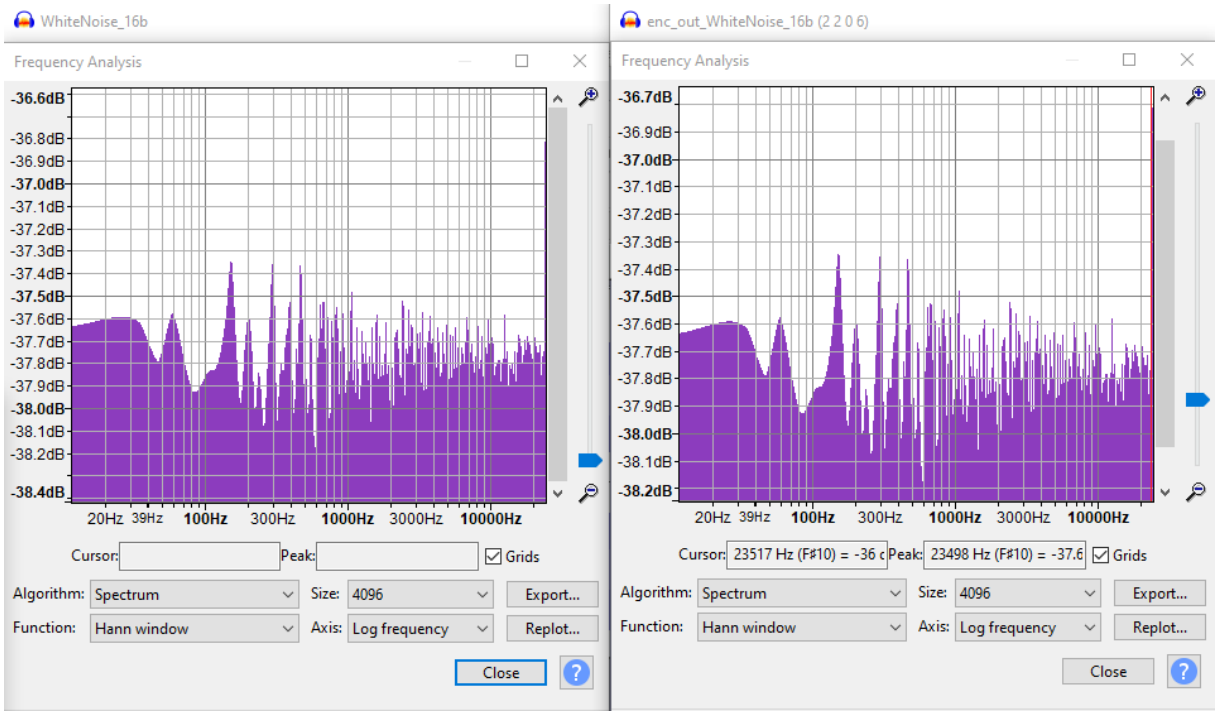
- да не постоји разлика у субјективној перцепцији сигнала који се добија као излаз из кодера и сигнала који се доводи на његов улаз и
- да се подаци који су у кодеру утиснути у аудио сигнал могу исправно реконструисати на декодерској страни, тј. да се на излазу декодера добија исправно обрађен (филтриран) сигнал

Да би се испитало да ли је први услов задовољен рађени су слушни тестови за неколико аудио токова у које су утиснути различити подаци, тј. рендеринг коефицијенти. Закључено је да нема разлике у субјективној перцепцији улазног и излазног сигнала. Такође је установљено и да нема видљивих разлика између спектралних карактеристика улазног и излазног сигнала. Последице утискивања информација у аудио ток се могу уочити само ако вршимо поређење на битском нивоу.

Слике 5.1 и 5.2 приказују поређење улаза и излаза кодера на битском нивоу и поређење њихових спектралних карактеристика. За поређење датотека на битском нивоу коришћен је алат *Compare By Content* у оквиру програма *Total Commander*, а за приказ спектралних карактеристика коришћен је програм *Audacity*.



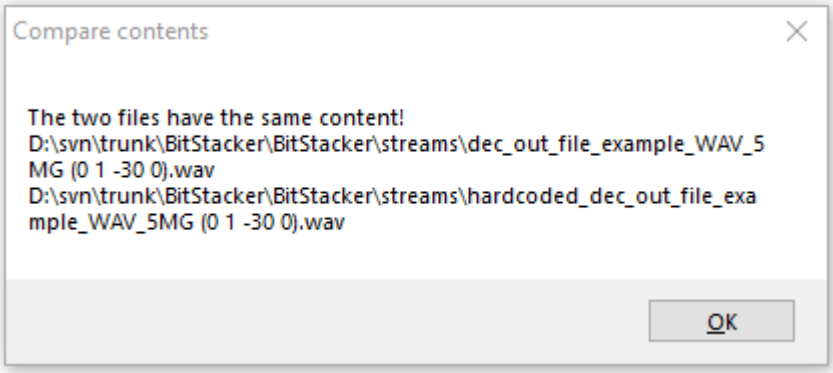
Слика 5.1 Поређење улаза и излаза кодера на битском нивоу



Слика 5.2 Поређење улаза и излаза кодера у спектралном домену

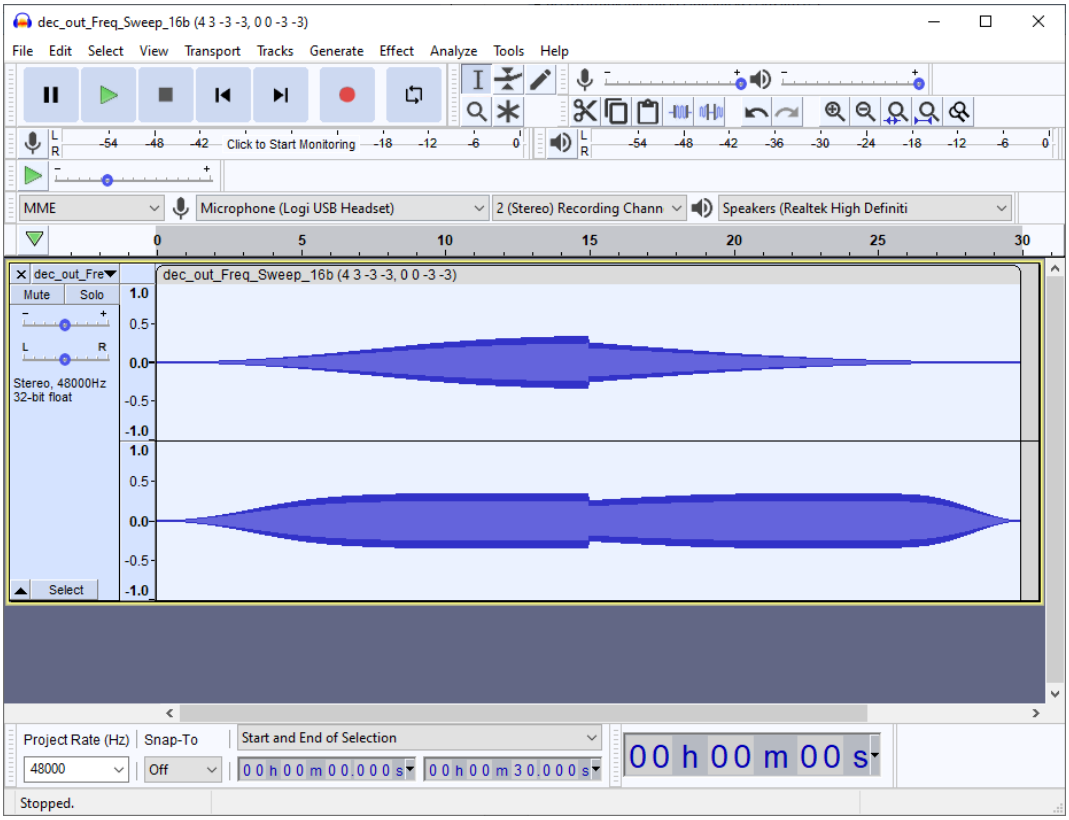
Са слике 5.1 се може видети да је максимална разлика између сваког од одбирака основног и енкодованог аудио тока 1 бит, на основу чега се и без слушних тестова могло закључити да људско ухо ту разлику неће моћи да осети.

Када је установљено да је први услов задовољен, тј. да не постоји разлика у субјективној перцепцији оригиналног звучног сигнала и звучног сигнала који садржи рендеринг коефицијенте, прешло се на испитивање другог услова. То је подразумевало проверу поклапања излаза из декодера који се добија када се на улаз доведе енкодовани сигнал са излазом из модификованог декодера који не врши реконструкцију енкодованих података, већ само обраду, која подразумева филтрирање. Филтрирање се, у модификованом декодеру, уместо на основу реконструисаних података из улазне датотеке, врши на основу претходно одређених вредности које треба да одговарају коефицијентима који су утиснути у сигнал у кодеру. Предефинисан је скуп тестова за поређење и генерисани су излази из модификованог декодера, а затим су ти излази на битском нивоу поређени са излазима из основног декодера и утврђено је да нема разлике између њих.



Слика 5.3 Поређење излаза из основног и модификованог декодера коришћењем алата *Compare By Content*

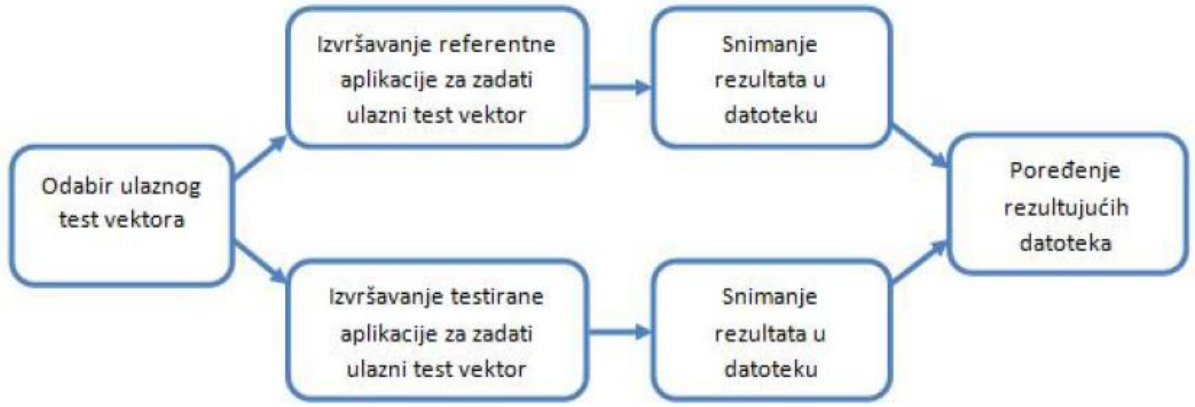
Тестирана је и могућност промене конфигурација. На слици испод (слика 5.3) је приказан филтрирани сигнал у временском домену. Прва половина аудио сигнала је филтрирана са конфигурацијама 4, 3, -3, -3, што значи да су граничне фреквенције пропусника опсега 8 kHz и 20 kHz, а улазно и излазно појачање су по -3 dB. Друга половина је филтрирана са конфигурацијама 0, 0, -3, -3, што значи да су граничне фреквенције пропусника опсега 2 kHz и 12 kHz, а улазно и излазно појачање су по -3 dB.



Слика 5.4 Приказ сигнала у временском домену

5.2 Испитивање осталих модела и симулаторске апликације

Према принципима развоја DSP апликација, о којима је било речи у поглављу 4.1, на основу референтног модела реализованог у *C* програмском језику креирани су остали модели. Било је очекивано да излази из ових модела одговарају излазима из референтног модела. Због тога је у овој фази било довољно само извршити бит-идентичне тестове. Одабран је скуп улазних тест вектора и креирана је .bat скрипта која покреће референтну и тестне апликације и као улазе им предаје тест векторе из поменутог скупа, а затим снимљене добијене излазе предаје *PCMCompare* алату за поређење, где се врши провера њиховог подударања.



Слика 5.5 Типичан сценарио испитивања [15]

Тестирање симулаторске апликације је вршено на исти начин као и тестирање модела из претходних фаза – коришћењем бит-идентичних тестова. Утврђено је да се добија очекивани резултат.

5.3 Потрошња ресурса

Након што је верификована исправност рада кодера за пренос рендеринг коефицијената извршена је процена утрошка ресурса. Извршена је процена утрошка меморије дигиталног сигнал процесора, као и процена утрошка процесорског времена потребног за обраду у кодерском модулу.

Мерење потрошње меморије циљне платформе се врши анализом датотеке са екстензијом .MAP која се добија приликом генерисања извршне датотеке. Посебно се посматрају меморија за податке (X и Y меморијска зона) и програмска меморија. Заузеће меморије се изражава у броју 32-битних речи. Потрошња меморије је приказана у табели испод (Табела 5.1).

X меморија	Y меморија	Програмска меморија
684	232	853

Табела 5.1 Потрошња меморије

Утрошак процесорског времена се мери у милионима инструкција по секунди (енгл. Million Instructions Per Second, MIPS) и израчунава се по следећој формули:

$$MIPS = \frac{broj\_ciklusa * \frac{Fs}{BLOCK\_SIZE}}{1000000}$$

Параметар *Fs* представља фреквенцију одабирања улазног сигнала, која овде износи 48 kHz, а параметар *BLOCK_SIZE* представља величину блока обраде, која је у овом случају 16 одбирака. Функционалност која се односи на бројање циклуса је интегрисана у симулатор. Измерен је број циклуса потребних за обраду једног блока од 16 одбирака. Када су сви параметри познати израчунат је утрошак процесорског времена, односно број *MIPS*-а.

Фреквенција одабирања	48 kHz
Број циклуса	867
MIPS	2.601

Табела 5.2 Потрошња процесорског времена

6. Закључак

Задатак овог рада је била реализација кодера за пренос рендеринг коефицијената заснованог на утискивању метаподатака у аудио податке на *DSP*. Како би се то постигло било је потребно упознати се са основним принципима и различитим методама аудио стеганографије. Такође је било потребно осмислити протокол за пренос података које је требало, применом неке од метода стеганографије, утиснути у аудио сигнал.

Метода аудио стеганографије која је примењена је метода кодовања бита најмањег значаја. Изабрана је због тога што није математички комплексна а омогућава утискивање велике количине информација у аудио. Мана ове методе је осетљивост на измене које могу настати приликом обраде сигнала. Међутим, у овом раду се обрада сигнала врши тек након декодовања па то не представља проблем.

Реализација кодера се одвијала кроз неколико фаза. Прво је креиран референтни *C* модел који се извршава на рачунару опште намене. За развој овог модела је коришћено интегрисано развојно окружење *Visual Studio*. Затим је, у наредним фазама, референтни модел прилагођаван тако да може да се преведе за циљну платформу. На основу *C* кода је, у развојном окружењу *CLIDE*, имплементран асемблерски код који је потпуно преводив за циљну платформу.

За потребе верфиковања решења, поред основног кодерског модула, креиран је и декодерски модул који је имплементиран у *C* програмском језику. Тестирање је вршено применом слушних, бит-идентичних и спектралних тестова.

Ово решење би се могло унапредити тако што би се повећао број *LSB* бита који се користи за кодовање и тиме повећала количина података које је могуће утиснути у аудио. Пре тога би било неопходно испитати колико бита се може изменити а да не

дође до деградације аудио сигнала. Решење би се могло модификовати и на тај начин да се избегне циклично слање низа пакета кроз цео аудио ток. Нови низ пакета би могао да се шаље само уколико се деси промена конфигурација које корисник задаје.

7. Литература

[1] "Introduction to Digital-Analog Conversion" [На мрежи]. Доступно: https://www.allaboutcircuits.com/textbook/digital/chpt-13/digital-analog-conversion. [Приступљено: 26.7.2022].

[2] K. Bhangale and M. Kothandaraman, "A review on speech processing using machine learning paradigm," *International Journal of Speech Technology*, vol. 24, no. 6, 2021.

[3] J. Lesurf, "MQA and "Bit-Stacking" ", 18.6.2016. [На мрежи]. Доступно: https://www.audiomisc.co.uk/MQA/bits/Stacking.html. [Приступљено: 22.7.2022].

[4] K. Saroha and P. K. Singh, "A Variant of LSB Steganography for Hiding Images in Audio," *International Journal of Computer Applications*, vol. 11, no. 6, 2010.

[5] "steganography" [На мрежи]. Доступно: https://www.merriam-webster.com/dictionary/steganography. [Приступљено: 18.07.2022].

[6] G. J. Simmons, "The Prisoners' Problem and the Subliminal Channel," in *Advances in Cryptology: Proceedings of CRYPTO '83*, Plenum, 1983, pp. 51-67.

[7] S. Kumar, BandyopadhyayBarnali and G. Banik, "LSB Modification and Phase Encoding Technique of Audio Steganography Revisited", *International Journal of*

Advanced Research in Computer and Communication Engineering, vol. 1, no. 14, 2012.

[8] G. Kipper, Investigator's Guide to Steganography, London: Auerbach Publications, 2004.

[9] G. Gori, "Tattooing as a vehicle for secret messages in ancient Greece" [На мрежи]. Доступно: https://www.tattoolife.com/tattooing-as-a-vehicle-for-secret-messages-in-ancient-greece/. [Приступљено: 21.7.2022].

[10] G. Chugh, "Image Steganography Techniques : A Review Article," vol. 6, no. 3, 2013.

[11] M. J. Surana, A. Sonsale, B. Joshi, D. Sharma and N. Choudhary, "Steganography Techniques," *International journal of Engineering development and research*, vol. 5, no. 2, pp. 989-992, 2017.

[12] P. Singh, "Types of Steganography methods that are used for hiding confidential data", 8.6.2022. [На мрежи]. Доступно: https://dev.to/prasan26/types-of-steganography-methods-that-are-used-for-hiding-confidential-data-1ebj. [Приступљено 20.7.2022].

[13] R. Olanrewaju, O. O. Khalifa and H. b. A. Rahman, "Increasing the hiding capacity of low-bit encoding audio steganography using a novel embedding technique", 2013.

[14] H. B. Kekre, A. Archana, R. Swarnalata and U. Athawale, "Information Hiding in Audio Signals," *International Journal of Computer Applications*, vol. 7, no. 9, 2010.

[15] J. Kovačević, D. Bokan, "Arhitekture i algoritmi digitalnih signal procesora: zbirka zadataka i laboratorijski priručnik," FTN izdavaštvo, Novi Sad, 2016.