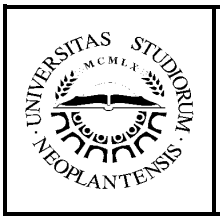


УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

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: Цветковић Катарина
Број индекса: РА166-2019

Тема рада: **Имплементација алгоритма управљања нискофреквентним садржајем за сложене аудио системе на ДСП платформи**

Ментор рада: доц. др Јелена Ковачевић

Нови Сад, јул 2023.

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**:		**Цветковић Катарина**
Ментор, **МН**:		**доц. др Јелена Ковачевић**
Наслов рада, **НР**:		**Имплементација алгоритма управљања нискофреквентним садржајем за сложене аудио системе на ДСП платформи**
Језик публикације, **ЈП**:		Српски / латиница
Језик извода, **ЈИ**:		Српски
Земља публиковања, **ЗП**:		Република Србија
Уже географско подручје, **УГП**:		Војводина
Година, **ГО**:		**2023.**
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)		**7/51/20/9/26/0/0**
Научна област, **НО**:		Електротехника и рачунарство
Научна дисциплина, **НД**:		Рачунарска техника
Предметна одредница/Кqучне речи, **ПО**:		**Дигитална обрада сигнала, Управљање нискофреквентним садржајем, Ефекти ниских фреквенција**
УДК		
Чува се, **ЧУ**:		У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, **ВН**:		
Извод, **ИЗ**:		**У оквиру рада имплементирано је једно решење ДСП модула за управљање нискофреквентним садржајем за сложене аудио системе на ДСП процесору фирме Cirrus Logic.**
Датум прихватања теме, **ДП**:		
Датум одбране, **ДО**:		21.7.2023.
Чланови комисије, **КО**:	Председник:	ванр. проф. др Иван Каштелан
	Члан:	доц. др Миодраг Ђукић — Потпис ментора
	Члан, ментор:	доц. др Јелена Ковачевић

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**:		**Cvetković Katarina**	
Mentor, **MN**:		**Jelena Kovačević, PhD**	
Title, **TI**:		**Implementation of a bass management algorithm for multichannel sound audio systems on a DSP platform**	
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**:		2023.	
Publisher, **PB**:		Author's reprint	
Publication place, **PP**:		Novi Sad, Dositeja Obradovica sq. 6	
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)		**7/51/20/9/26/0/0**	
Scientific field, **SF**:		Electrical Engineering	
Scientific discipline, **SD**:		Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, **S/KW**:		Digital signal processing, Bass Management, LFE(Low Frequency Effect) channel	
UC			
Holding data, **HD**:		The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, **N**:			
Abstract, **AB**:		This paper describes implementation of a bass management solution for multichannel audio systems with subwoofers for Cirrus Logic digital signal processor.	
Accepted by the Scientific Board on, **ASB**:			
Defended on, **DE**:		21.7.2023.	
Defended Board, **DB**:	President:	Ivan Kaštelan, PhD	
	Member:	Miodrag Đukić, PhD	Menthor's sign
	Member, Mentor:	Jelena Kovačević, PhD	

Захвалност

Захваљујем се Институту РТ-РК на могућности реализације дипломског рада, менторки, доц. др Ковачевић Јелени, техничкој менторки, Савић Јасмини, као и Обуђа Азри на стручној помоћи, корисним саветима и доступности.

Захваљујем се породици и пријатељима на константној подршци током школовања, као и колегама са одсека на свакој помоћи и бодрењу у претходним годинама студирања.

САДРЖАЈ

1. Увод 10
2. Репродукција нискофреквентног садржаја у сложеним аудио система 12
2.1 Развој аудио система у кућним биоскопима 12
2.2 Аудио/Видео пријемник 15
2.3 Soundbar 16
2.4 Управљање нискофреквентним садржајем – Bass Management 17
2.5 LFE канал и његове карактеристике 20
3. Развојно окружење и концепт решења 23
3.1 Карактеристике дигиталног сигнал процесора 23
3.2 Методологија развоја ДСП апликација 25
3.2.1 Модел 0 - референтни код 27
3.2.2 Модел 1 - функционална оптимизација С кода 28
3.2.3 Модел 2 - прилагођење кода аритметици ДСП-а 28
3.2.4 Модел 3 - превођење кода компајлером за циљну архитектуру 28
3.2.5 Финални модел – код извршив на циљној платформи 29
3.3 Развојно окружење 29
3.3.1 Програмски преводилац CCC2 30
3.4 Програмско окружење Cirrus Logic Framework 31
4. Програмско решење 33
4.1 Оптимизације током израде Модела 3 34
4.1.1 Оптимизација програмских петљи 34
4.1.2 Оптимизација условних исказа 34
4.1.3 Оптимизација приступа меморији 35

4.1.4 Коришћење кружног бафера ... 35
4.2 Специфичности рада са компајлером CCC2 ... 35
4.3 Функција remap_input ... 36
4.4 Функција module_init ... 36
4.5 Функција processing ... 36
4.5.1 Функција processing_iir ... 37
4.5.2 Функција processing_delay_gain ... 37
4.6 Функција remap_outputs ... 37
5. Испитивање и резултати ... 38
5.1 Аутоматизација процеса испитивања ... 39
5.2 Утрошак ресурса у модулу за управљање нискофреквентним садржајем 41
5.3 Верификација рада на дигиталном сигнал процесору ... 44
5.3.1 Слушно тестирање ... 44
5.4 Поређење резултата са асемблерском имплементацијом модула ... 45
6. Закључак ... 47
7. Литература ... 49

Списак слика

Слика 1.1 Пример поставке система за калибрацију 11
Слика 2.1 Основни елементи кућног A/V система [3] 13
Слика 2.2 Пример ланца декодерске обраде у систему за кућне биоскопе [4] 14
Слика 2.3 Пример завршне аудио обраде у систему за кућне биоскопе [4] 14
Слика 2.4 Приказ изгледа и блок дијаграм пријемника аудио/видео сигнала заснованог на ДСП процесору CS49x 15
Слика 2.5 Типични систем кућног биоскопа [3] 16
Слика 2.6 Илустрација пропагирања директног звука с предњих звучника и рефлексије звука горњих звучника у саундбар систему [11] 17
Слика 2.7 Систем са сабвуферима 18
Слика 2.8 Систем без сабвуфера 19
Слика 2.9 Систем са ограничавањем нивоа садржаја 19
Слика 2.10 Излаз мерења фреквенцијског одзива са једним сабвуфером 21
Слика 2.11 Излаз мерења фреквенцијског одзива са више сабвуфера 21
Слика 2.12 Пример 5.1 конфигурације звучника 22
Слика 3.1 Блок дијаграм процесора CS49844 23
Слика 3.2 Блок дијаграм DSP језгра Crystal 32 25
Слика 3.3 Ток имплементације софтвера на ДСП процесорима са аритметиком непокретног зареза 27
Слика 3.4 Контролисано извршавање програма у CLIDE окружењу 30
Слика 3.5 Дијаграм тока превођења изворног кода 30
Слика 3.6 Спрега модула са ОС-ом 32
Слика 4.1 Блок шема алгоритма за управљање нискофреквентним садржајем 33
Слика 5.1 Испис аутоматизованог испитивања у конзоли python скрипте 40

Слика 5.2 Пример текстуалне датотеке након покретања скрипте..... 40
Слика 5.3 Пример ручног поређења излаза 5.1 аудио система 40
Слика 5.4 Пример ручног поређења излаза 5.2 аудио система 41
Слика 5.5 Пример конфигурације модула и покретања ДСП апликације 44
Слика 5.6 Окружење коришћено током извођења слушних тестова..... 45

Списак табела

Табела 3.1 Расположива меморија процесора CS49844 24
Табела 5.1 Садржај тестног вектора input_0x00009f.wav 38
Табела 5.2 Карактеристике имплементације за 11.2 аудио систем 42
Табела 5.3 Утрошак MIPS-а пре оптимизације кода за аудио систем 11.2 42
Табела 5.4 Утрошак MIPS-а након оптимизације кода за аудио системе 11.2 43
Табела 5.5 Карактеристике имплементације за 5.2 систем 43
Табела 5.6 Утрошак MIPS-а након оптимизације кода за аудио систем 5.2 43
Табела 5.7 Потрошња меморије (број речи) за 5.2 систем 44
Табела 5.8 Поређење утрошка ресурса C и асемблерског решења за 5.1 конфигурацију 46

Скраћенице

DSP - *Digital Signal Processing*, Дигитална обрада сигнала

LFE - *Low-Frequency Effects*, Ефекти ниских фреквенција

AVR - *Audio/Video Receiver*, Аудио/видео пријемник

MAC - *Multiply and Accumulate*, Помножи и додај

OS - *Operating System*, Оперативни систем

MIF - *Module Interface*, Спрежни подсистем модула

MCT - *Module Call Table*, Табела рутина за спрегу са ОС-ом

MCV - *Module Control Vector*, Табела конфигурационих параметара

ODT - *Overlay Definition Table*, Табела која садржи показиваче на MIF табеле свих учитаних модула

CLIDE - *Cirrus Logical Integrated Development Environment*, Cirrus Logic интегрисано развојно окружење

CCC - *Cirrus C Compiler*, Cirrus C компајлер

ALU - *Arithmetic and Logical Unit*, Аритметичко логичка јединица

SRS - *Shifter/Rounder/Saturation*, Померање, заокруживање, засићење

IIR - *Infinite Impulse Response*, Бесконачан импулсни одзив

MIPS - *Million Instruction Per Second*, Милиони инструкција по секунди

HDMI - *High-Definition Media Interface*, Спрега за мултимедијални садржај

S/PDIF - *Sony/Phillips Digital Interface*, Стандард који прописује повезивање аудио уређаја на кратким растојањима, дефинисан од стране компанија Sony и Phillips

SPI - *Serial Peripheral Interface*, Серијски спрежни систем за периферије

I²C - *Inter-Integrated Circuit*, Серијска магистрала која омогућава спрезање више водећих и више пратећих уређаја (интегрисаних кола)

I²S - *Inter-IC Sound*, Серијски протокол развијен за дигитални пренос аудио података између интегрисаних кола

PCM - *Pulse-code modulation*, Импулсна кодна модулација

1. Увод

Кућни биоскопи (енг. *Home-Theater*) представљају конфигурацију аудио и видео опреме у кућним условима која опонаша искуство правих биоскопа, чиме гледање телевизије и филмова код куће постаје занимљивије и узбудљивије. Угођај који пружају кућни биоскопи не би били потпуни без посебних звучних ефеката који долазе са звучника који се шире целом просторијом. Међутим, да би се дошло до савршеног угођаја код слушаоца није довољно само прикључити звучнике на систем за репродукцију аудио/видео садржаја и појачати их. Фреквенције свих нивоа захтевају усмеравање на тачно одређене излазне канале као и повезивање на одговарајуће звучнике како би аудио садржај био што верније репродукован. Један од потребних модула завршне обраде аудио сигнала за решавање овог проблема у нискофреквентном домену је модул за управљање нискофреквентним (енг. *bass – бас*) садржајем (енг. *Bass Management*).

Аудио системи за кућне биоскопе међусобно се разликују по броју звучника који варирају од обичних стерео система (леви и десни предњи звучник) до сложенијих система (нпр. систем 5.1 се састоји од 5 главних звучника пуног опсега и 1 ограниченог опсега). Да би се садржај оптимално репродуковао на сваком систему потребно је извршити различите врсте завршне обраде. Један од проблема који треба решити је управљање нискофреквентним садржајем услед карактеристика самих звучника доступних звучника у систему, као и карактеристика саме просторије у којој се садржај репродукује.

Задатак овог дипломског рада јесте развој сложеног модула за управљање нискофреквентним садржајем (енг. LFE – *Low Frequency Effect*) на крајњем уређају

(аудио дигитални сигнал процесор фирме Cirrus Logic) који врши обраду на основу података (конфигурација звучника, коефицијенти филтара итд.) добијених од корисничке апликације за калибрацију аудио система, ослањајући се на методологију за развој ДСП апликација и језичка проширења С програмског језика за писање ДСП апликација.

Слика 1.1 Пример поставке система за калибрацију

Коефицијенти добијени у процесу калибрације аудио система помоћу корисничке апликације на рачунару биће искоришћени за подешавање модула за управљање нискофреквентним аудио садржајем на ДСП-у (Слика 1.1).

2. Репродукција нискофреквентног садржаја у сложеним аудио система

2.1 Развој аудио система у кућним биоскопима

До касних '70-их година, кућни биоскопи подржавали су само моно (једноканални) звук. Представљањем Dolby Stereo звука 1975. године, претходна технологија је превазиђена, док је стерео звук кроз 2 године постао општеприхваћен приказивањем филма "Ратови звезда" 1977. године.

На простору предвиђеном за конвенцијалну моно траку нашле су се две звучне траке које, поред информација о левом и десном каналу, такође могу помоћу матричног процеса енкодовања да садрже и трећи, односно централни, и четврти просторни (енг. *surround*) канал за специјалне ефекте и амбијентални звук. Иако је тада изашао на тржиште, Dolby Surround је тек деценију касније прилагођен за репродукцију у кућним биоскопима. У понуди је постојао само један аудио видео пријемник (енг. *Audio Video Receiver* – AVR) са уграђеним Dolby Surround декодером и који је имао самосталне Dolby процесоре који су могли бити убачени у постојеће системе како би стварали просторни звук. Dolby Surround је тада био систем четири канала: десни и леви предњи, централни канал за дијалог и један просторни канал који је био подељен и распоређен на један пар просторних звучника.

Са друге стране, Dolby Pro Logic, представљен 1987. године, није био другачијег формата, али је имао унапређену методу обраде постојећих звучних трака. Декодер ове технологије користио је логику управљања за повећање и смањење јачине сваког канала независно, и тиме омогућавајући прецизније постављање дијалога, музике и ефеката.

И Dolby Surround и Dolby Pro Logic биле су аналогне аудио технологије које су покушале максимално да искористе дводимензионални звук канала, тако што су се

четири канала звука спајала у два током енкодовања, да би се на крајњем уређају отпаковала у четири и тако репродуковала.

Прва дигитална технологија која се затим развила била је Dolby Digital, која је користила дигитални аудио како би створила просторни звук са шест одвојених, тј. „дискретних" канала: леви и десни предњи (енг. *front*), централни, независни леви и десни просторни канали, и додатни канал – LFE за пренос нискофреквентних звучних ефеката. Овакав систем обезбедио је велико унапређење у целокупном квалитету звука, са напретком у издвајању чистог звука од шума, као и много прецизније просторно распоређивање самог звука. Аудио систем 5.1 је и даље најпопуларнија поставка просторног звука [1].

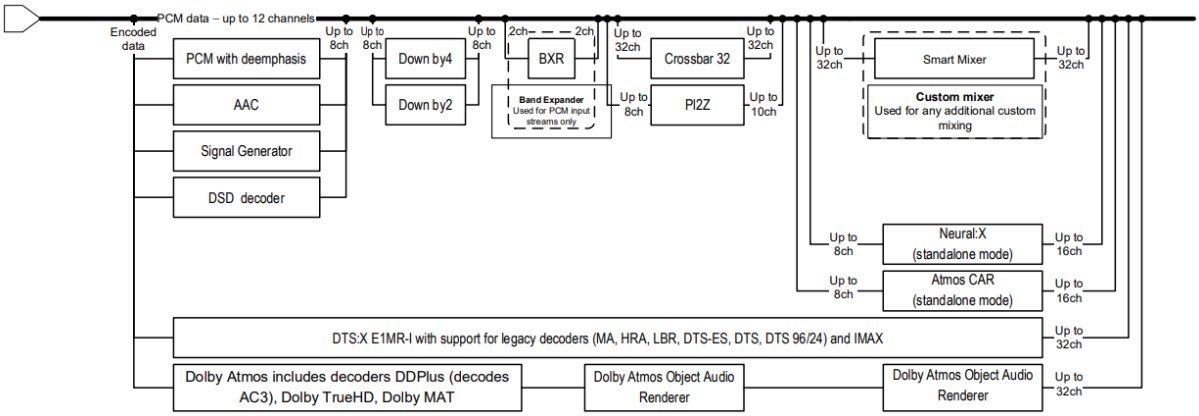
Даљим развојем, технологије просторног звука су, осим дводимензионалног (2Д) просторног садржаја – где слушалац перципира изворе звука само у хоризонталној равни, омоућиле и тродимензионалну (3Д) репродукцију аудио садржаја [2].

Слика 2.1 Основни елементи кућног A/V система [3]

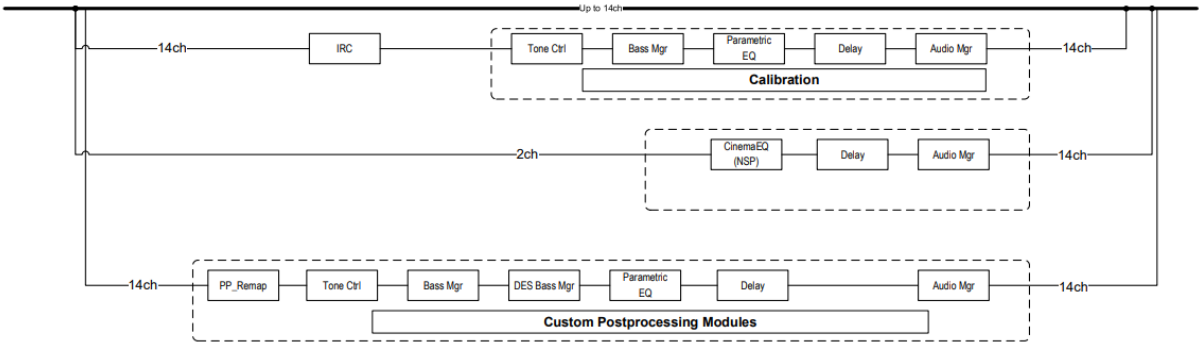
Типичан систем за кућне биоскопе састоји се од извора, разводника и одредишта (Слика 2.1). Извор је уређај који има способност да прими аудио/видео сигнал (преко мреже, са физичког медија) и проследи га до разводника одређеном аудио/видео спрегом (HDMI, S/PDIF). Разводник има улогу да декодује (и по могућности обради) сигнал и проследи га одговарајућом аудио/видео спрегом до одредишта (звучници, екран).

Системи за кућне биоскопе обично подржавају репродукцију засновану на више комплексних аудио технологија (Dolby Digital, Dolby Atmos, DTS:X, MPEG-H, AAC,

PCM), укључујући и додатне завршне обраде ради што верније репродукције аудио садржаја (Слика 2.2, Слика 2.3).



Слика 2.2 Пример ланца декодерске обраде у систему за кућне биоскопе [4]



Слика 2.3 Пример завршне аудио обраде у систему за кућне биоскопе [4]

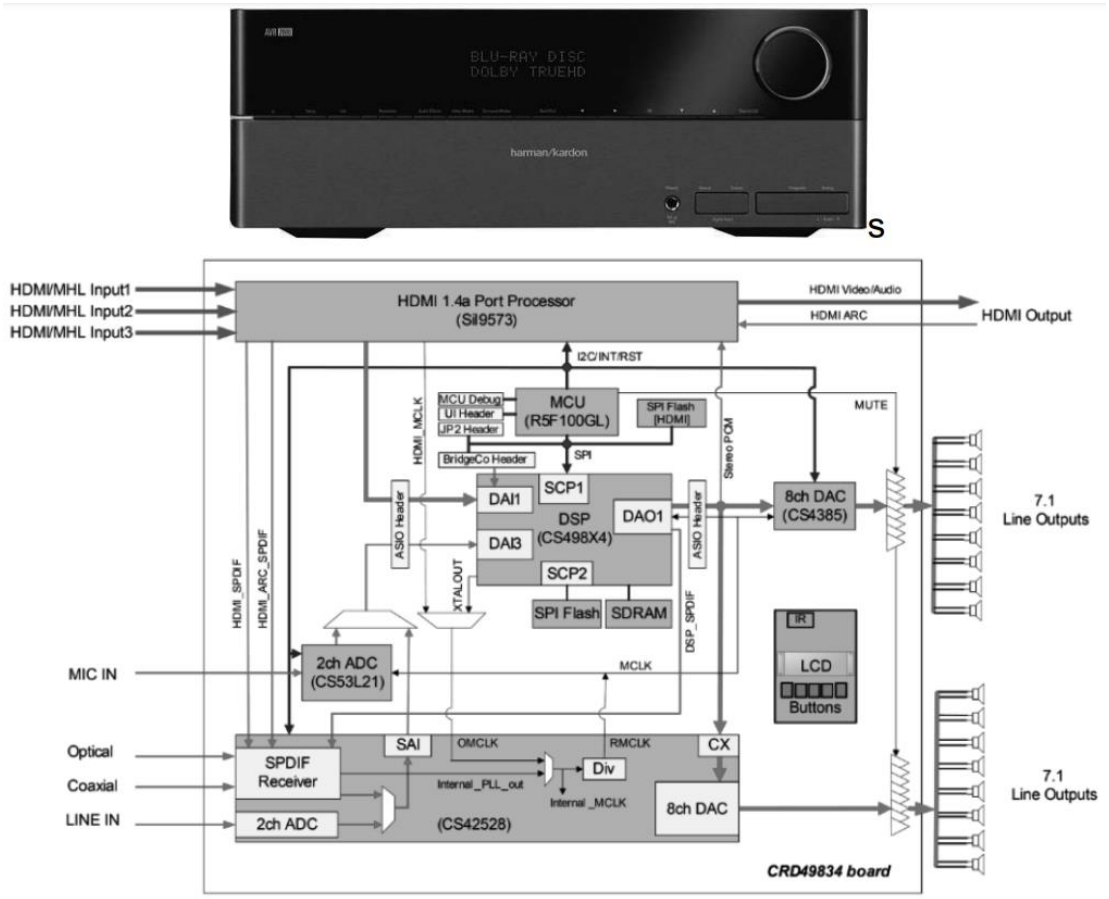
Према истраживању компаније Technavio, за тржиште система за кућне биоскопе предвиђен је раст од 7.4 милијарди америчких долара између 2022. и 2027. године, где се као главни фактор наводи приступачност ових уређаја услед веће куповне моћи (и очекивања) потрошача. Овакав раст може остати одржив само уколико се смањи потрошња производње самих уређаја, јер куповна моћ потрошача не прати растуће цене ових уређаја [5].

У уређајима потрошачке електронике, као што су системи за кућне биоскопе, потребно је узети у обзир и цену компоненти које чине тај уређај и потрошњу енергије. Дигитални сигнал процесори за обраду дигиталног аудио сигнала се овде показују као задовољавајуће решење. Они омогућавају решење на једном чипу (енг. *single-chip solution*) за обраду комплексних аудио технологија, где је аудио обрада изолована од остатка система. Додатно, због њихове високе прилагођености обради аудио сигнала у виду програмске подршке, архитектуре и скупа инструкција, омогућавају извршавање комплексних алгоритама декодовања и аудио обраде, уз мањи утрошак програмских

ресурса, што омогућава производњу дигиталних сигнал процесора са веома малим утрошком енергије, а самим тим и нижом ценом.

2.2 Аудио/Видео пријемник

Пријемник аудио и видео садржаја има улогу да прими сигнал на улазу, који може бити аналогни или дигитални, декодује га (по потреби обради), и тако обрађен сигнал да претвори у аналогну величину, која се даље води на аудио појачавач. Излаз из аудио појачавача представља сигнал за побуду звучника. Један пример аудио/видео пријемника заснованог на ДСП решењу приказује Слика 2.4.



Слика 2.4 Приказ изгледа и блок дијаграм пријемника аудио/видео сигнала заснованог на ДСП процесору CS49x

ДСП комуницира са периферним уређајима преко SPI или I²C магистрала. У овом систему, ДСП је SPI магистралом повезан са екстерном флеш меморијом, у којој је смештен тзв. *image* фајл са целокупним ДСП фирмвером (декодерима, пост-процесорима) и свим подешавањима потребним за рад ДСП-а. Кључну улогу у AVR систему има микроконтролер који је одговоран за конфигурацију и правила рад система (HDMI примопредајник, S/PDIF пријемник), пуштање у рад ДСП-а, као и управљањем целокупним радом система, реаговање на грешке, итд [6].

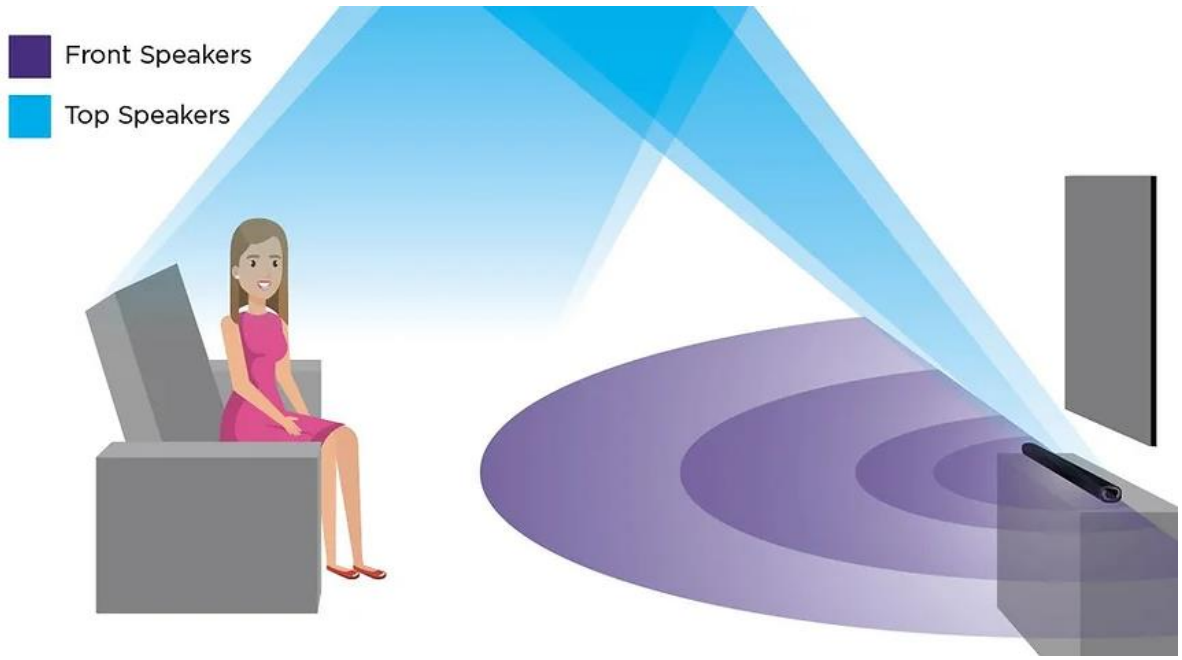
Један типичан пример система за кућне биоскопе дат је на Слика 2.5.

Слика 2.5 Типични систем кућног биоскопа [3]

2.3 Soundbar

Прве саундбарове (енг. *soundbar*) увела је Altec Lansing Corp. 1998. године [7] са циљем искоришћења више звучника пуног опсега и звучника ограниченог опсега, тзв. „малих“ звучника заједно са технологијом која усмерава звук ради контролисане рефлексије звука за симулацију и репродукцију просторних ефеката. Данас скоро сви водећи произвођачи аудио система нуде различите врсте саундбарова од више звучника у једној кутији саундбара до решења са више кутија саундбара [8][9]. Најнапреднији уређаји користе предње, задње, просторне и горње звучнике заједно са одговарајућим ДСП процесором са технологијом усмереног снопа (енг. *beamforming*) – помоћу које се звучни сигнал усмерава ка зидовима просторије ради контролисане рефлексије звучних сигнала [10].

Један од главних акустичних принципа у основи саундбар технологија је позиција звучника која помоћу технологије усмереног снопа формира акустичну рефлексију о зидове (код напредних система и плафон) у просторији до слушаоца (Слика 2.6).



Слика 2.6 Илустрација пропагирања директног звука с предњих звучника и рефлексије звука горњих звучника у саундбар систему [11]

2.4 Управљање нискофреквентним садржајем – Bass Management

Сабвуфер (енг. *subwoofer*) звучници (звучници резервисани за репродукцију ниских фреквенција) имају способност да репродукују садржај било ког канала у систему. Избор канала чији ће нискофреквентни садржај бити репродукован на сабвуферу врши модул за управљање нискофреквентним садржајем. На пример, поред нискофреквентног садржаја из LFE канала, сабвуфер може репродуковати и нискофреквентни садржај из централног и просторног канала, у случају да њима кореспондентни звучници нису у стању да адекватно репродукују ниске фреквенције.

Термин управљање нискофреквентним садржајем појавио се средином '90-их година прошлог века. Искоришћен је како би описао низ комплексних и напредних функција у AVR-у, које помажу кориснику током конфигурисања аудио-видео система, са акцентом на што бољу репродукцију нискофреквентног садржаја.

Концепт управљања нискофреквентним садржајем темељи се на постојању:

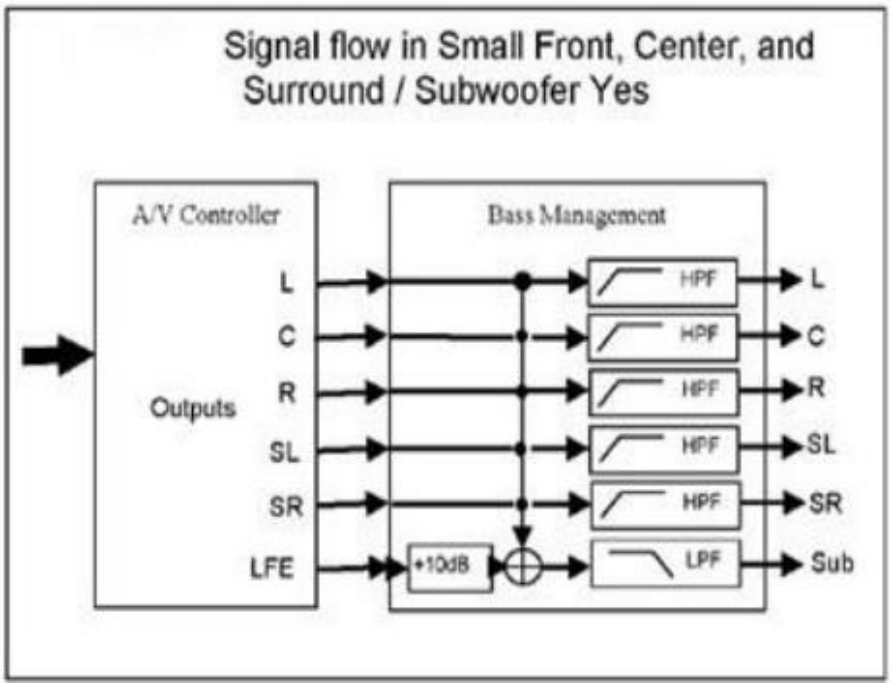
- система за одређивање фреквенцијског опсега сваког звучника у систему,
- фреквенцијски прилагодљивих филтара за сваки звучник у систему, како би се обезбедило да се на звучник шаље само део спектра сигнала који је могуће репродуковати,
- система за ограничавање амплитуде нискофреквентног садржаја, како би се спречило преоптерећење звучника,

- сложеног система за преусмеравање нискофреквентног садржаја из свих канала,
- система за процену одзива просторије и евентуалну корекцију поменутих филтара.

Међутим, овај приступ се не користи у комерцијалним уређајима јер и мање софистицирана решења дају задовољавајуће карактеристике.

Код свих модерних аудио-видео уређаја мора да постоји одређени облик управљања нискофреквентним садржајем који укључује високопропусне филтре за уклањање нискофреквентног садржаја и слање само високих фреквенција на све главне канале, као и нископропусне филтре за извлачење дела ниских фреквенција улазног сигнала и њихово спровођење до сабвуфера. Ови филтри треба да буду конфигурисани на основу два фактора:

- способности главних звучника да репродукују садржај из нискофреквентног опсега и
- облика просторије у којој ће систем звучника бити постављен.

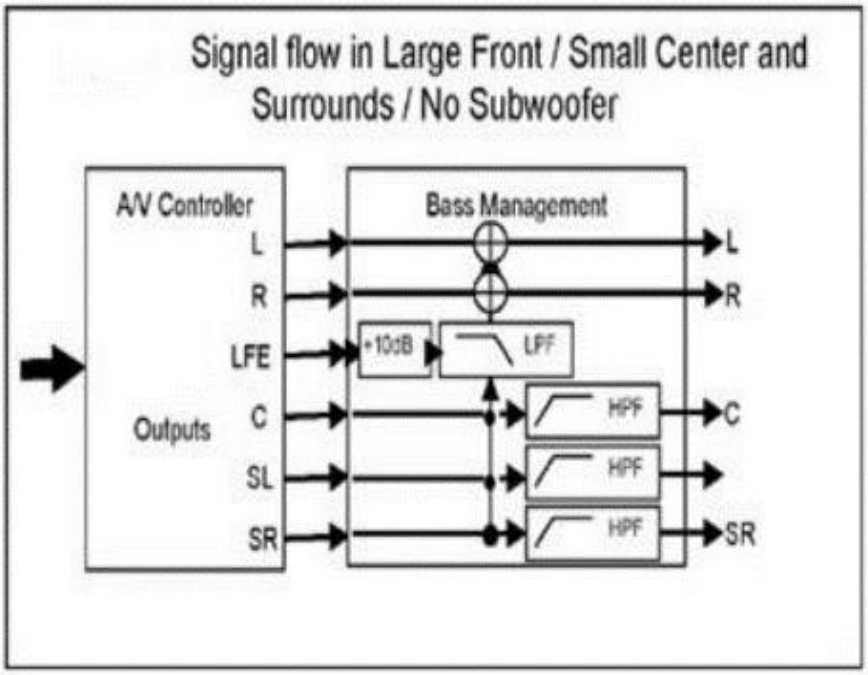


Слика 2.7 Систем са сабвуферима

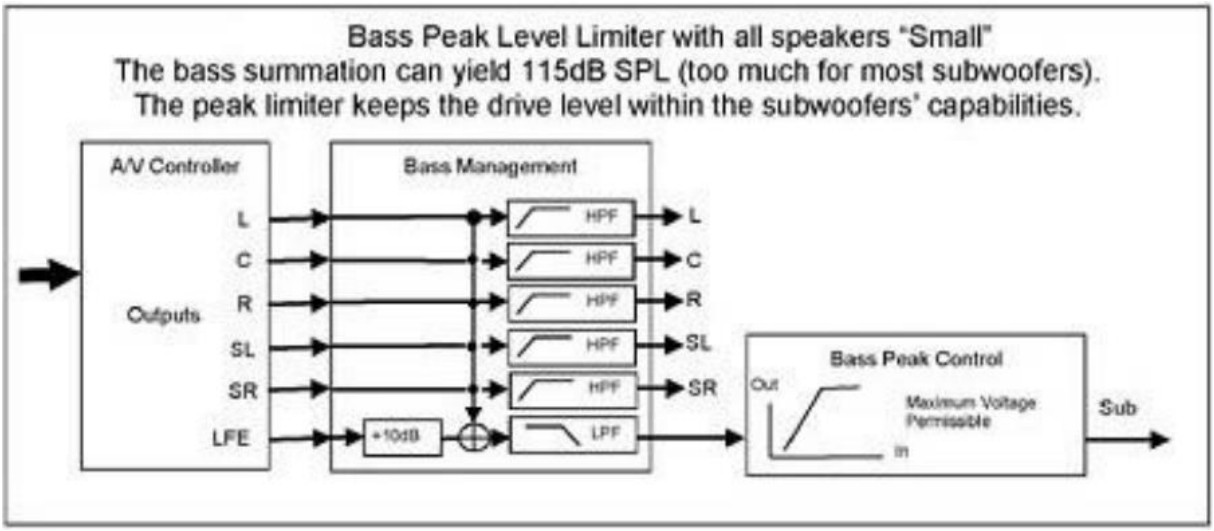
Поред тога, треба разматрати и преусмеравање нискофреквентног садржаја који би требало да је могуће, у идеалним условима, преусмерити из било ког канала у било који други канал, на произвољном нивоу и произвољној фреквенцији. У зависности од тога да ли садрже сабвуфере или не, системи различито преусмеравају свој садржај. На следећим сликама представљени су примери оба система. Слика 2.7 приказује систем са сабвуферима, где се сав нискофреквентни садржај из предњих, централног и просторних

канала заједно са садржајем LFE канала шаље у сабвуфере. Понекад систем нуди опцију да се нискофреквентни садржај из „малих" звучника, који га не могу репродуковати, шаље и на „велике" звучнике и на сабвуфере. То није пожељно, из разлога што може доћи до интерференције. Слика 2.8 је представа система без сабвуфера, у којем се нискофреквентни садржај из централног, просторног и LFE канала шаље у предње звучнике [12].

На крају, када су високопропусни и нископропусни филтри конфигурисани и комплетан нискофреквентни садржај распоређен на одговарајуће звучнике, у систем се укључује ограничавање нивоа тог садржаја. Улога ове обраде је да спречи превисок ниво излазног напона који би могао преоптеретити звучнике или сабвуфере.



Слика 2.8 Систем без сабвуфера



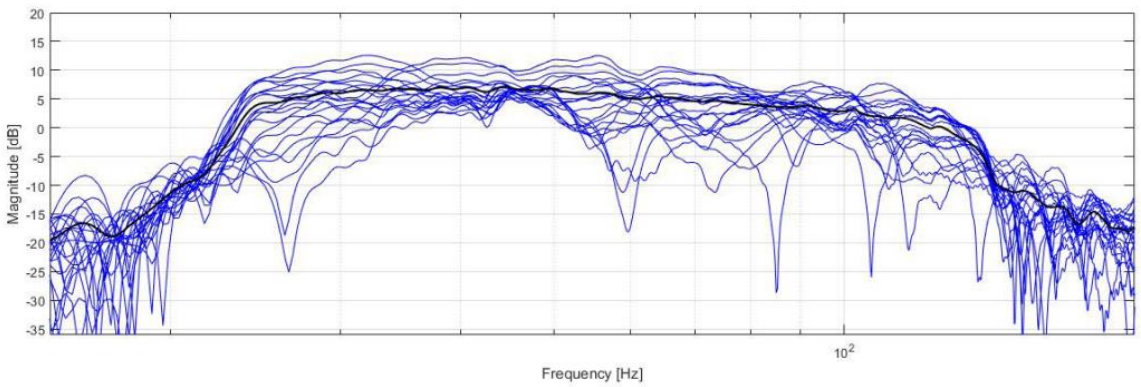
Слика 2.9 Систем са ограничавањем нивоа садржаја

2.5 LFE канал и његове карактеристике

За разлику од главних канала, LFE канал носи једино нискофреквентне информације које су мање од 120Hz . Првобитно је осмишљен за Dolby Stereo 70mm филмске траке како би репродуковао одвојени нискофреквентни сигнал на један или више сабвуфера смештених иза екрана. То је омогућавало јасне нискофреквентне ефекте у филмовима без потребе за додатним усложњавањем постојећих звучника и појачала код главних канала. Средином '90-их и '00-их постао је уобичајена појава у кућним биоскопима [13].

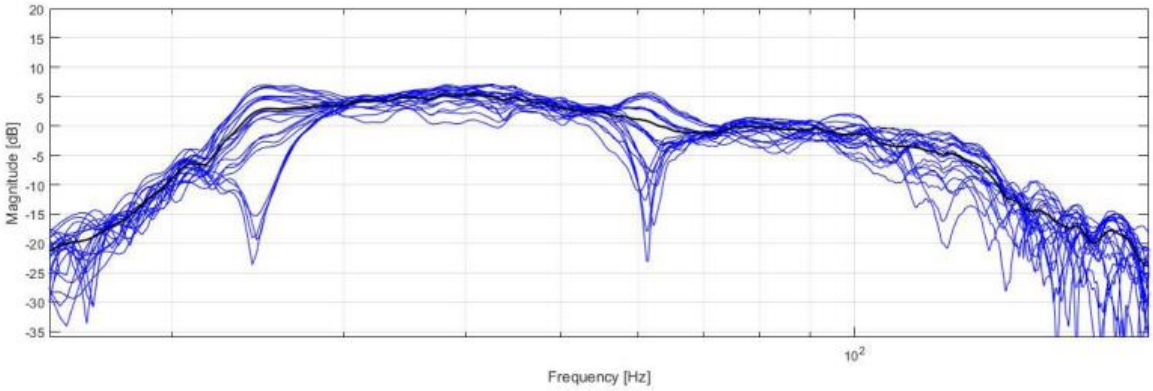
LFE канал носи додатне нискофреквентне садржаје који могу допунити нискофреквентни садржај главних канала. Сигнал у LFE каналу је калибрисан током снимања и омогућује 10dB виши ниво звучног притиска (енг. *Sound Pressure Level* – SPL) него исти нискофреквентни сигнал у неком од основних канала. Управо ова особина омогућила је филмским стваралоцима да интензивне нискофреквентне сигнале попут звука експлозије, земљотреса, итд. сместе у LFE канал [12].

Садржај LFE канала се најчешће шаље на звучнике који су наменски дизајнирани за репродукцију оваквих садржаја, тзв. сабвуфере. Сабвуфери код јефтинијих система обично преузимају садржај ниских фреквенција до 200Hz због ограничености главних звучника, док је код професионалних система та граница нижа и обично износи око 80Hz. Први сабвуфери су развијени у '60-им годинама за кућне стерео системе, али су постали нарочито популарни са појавом филмова који су садржали нискофреквентне ефекте намењене за репродукцију на сабвуфер звучницима. Сабвуфер може бити део система звучника који садржи мале звучнике „сателите" или може бити уграђен у исто кућиште са основним звучницима (нпр. „*tower*" звучници). Код комплета звучника за кућне биоскопе популарнији су системи који садрже физички одвојене сабвуфере и „сателите". Како сваки аудио систем тежи да има што бољи фреквенцијски одзив који неће бити лако променљив са променом позиције слушаоца у просторији, важно је напоменути да број активних сабвуфера игра важну улогу у овом случају. Одређене неправилности се могу појавити уколико главни звуци доминирају над ниским фреквенцијама тако да фреквенције драматично варирају у односу на позицију слушаоца у просторији. Слика 2.10Слика 2.10 приказује пример фреквенцијског одзива на сабвуферу измереног на 21 позицији у просторији код система који садржи само један сабвуфер. Иако је видљиво да се у просеку све фреквенције добро понашају (црна линија), ипак се јављају и бројне неправилности са променама позиција које варирају од 20dB до 30dB у одређеним тачкама.



Слика 2.10 Излаз мерења фреквенцијског одзива са једним сабвуфером

На основу мерења, закључено је да коришћење више сабвуфера помаже у смањењу неправилности, поготово уколико се позиција и однос фаза пажљиво изаберу тако да на оптималном нивоу комуницирају међусобно, а и са самом просторијом. Друга ствар коју подстиче коришћење више сабвуфера јесте повећање јачине нискофреквентног садржаја с обзиром да се потребна електрична и механичка снага тада распоређује преко више звучних елемената. Слика 2.11 приказује резултате мерења фреквентног одзива акустичне суме три активна сабвуфера на 21 позицији у просторији. Поређењем са претходним мерењем, закључујемо да је за смањење неправилности видно значајан број сабвуфера, али да ипак не можемо у потпуности предвидети резултат тако да се све неправилности у потпуности реше [14].



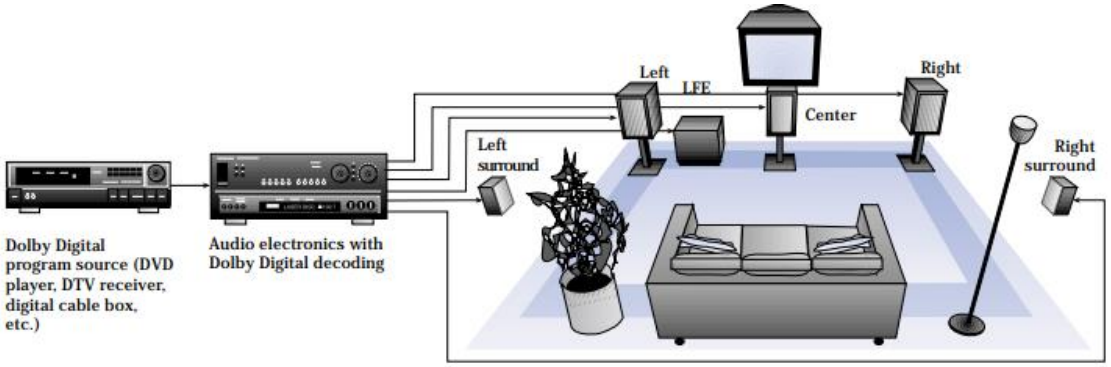
Слика 2.11 Излаз мерења фреквенцијског одзива са више сабвуфера

Често се LFE канал поистовећује са сабвуфером, међутим термини LFE канал и сабвуфер нису синоними. Могуће је да програм садржи LFE канал, али да декодер не генерише сабвуфер излаз пошто се сав нискофреквентни садржај основних канала као и садржај LFE канала може репродуковати са постојећим звучницима. И обрнуто такође може да се деси, да емитовани програм не садржи LFE канал, али да декодер генерише

сабвуфер излаз пошто поједини или чак сви звучници у систему нису способни да репродукују нискофреквентни садржај.

Разлика између LFE канала и сабвуфера је та што LFE преноси додатне нискофреквентне садржаје емитованог програма, а сабвуфер звучник представља начин, тј. компоненту на којој се нискофреквентни садржај репродукује и зато их не треба мешати.

У сложеним аудио системима обично је неколико звучника пуног опсега (енг. *full range*) и један или више звучника ограниченог опсега (енг. *band-limited*). Овакви системи се означавају нотацијом x.y, где x представља број канала пуног опсега, а y број LFE канала. Најпопуларнији аудио систем 5.1 садржи пет канала пуног опсега и један LFE канал. Слика 2.12 приказује пример 5.1 система за репродуковања Dolby Digital садржаја.

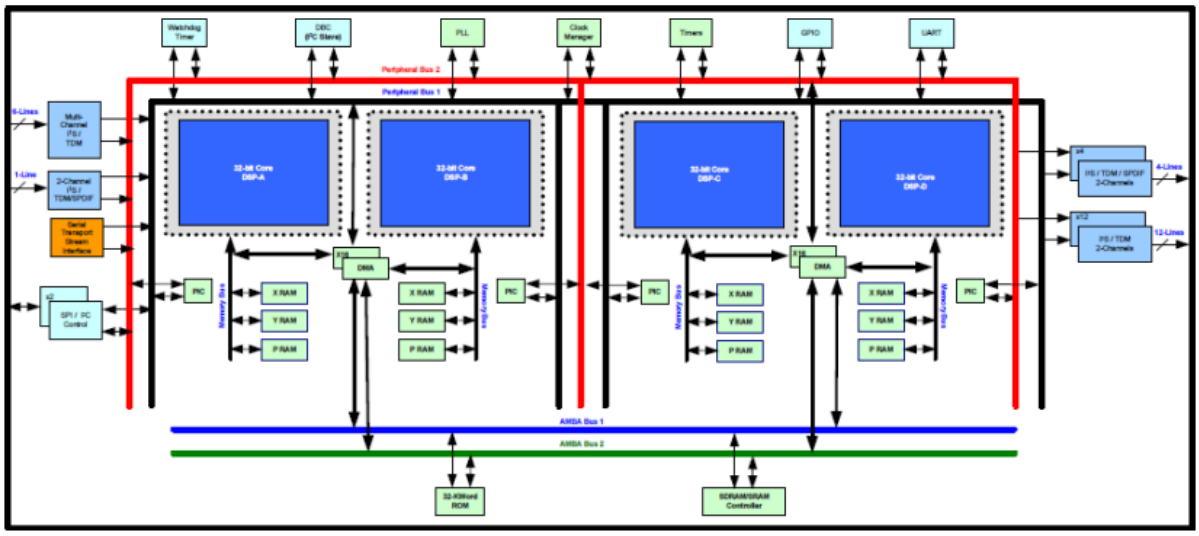


Слика 2.12 Пример 5.1 конфигурације звучника

3. Развојно окружење и концепт решења

У овом поглављу биће описани платформа, развојно окружење и ток израде пројектног задатка.

3.1 Карактеристике дигиталног сигнал процесора



Слика 3.1 Блок дијаграм процесора CS49844

Развој сложеног модула за управљање LFE садржајем извршава се на крајњем уређају ДСП (енг. DSP – Digital Signal Processor) процесора фамилије CS498xx компаније Cirrus Logic, који је намењен обради аудио сигнала у уређајима потрошачке електронике. Првенствено се уграђују у AVR и активне звучнике, али имају и примену у телевизорима, аутомобилским аудио системима и преносним уређајима. Засновани су на архитектури ДСП језгра под именом Crystal 32, а разлике између модела унутар фамилије се односе пре свега на број језгара, радни такт и количину интерне меморије.

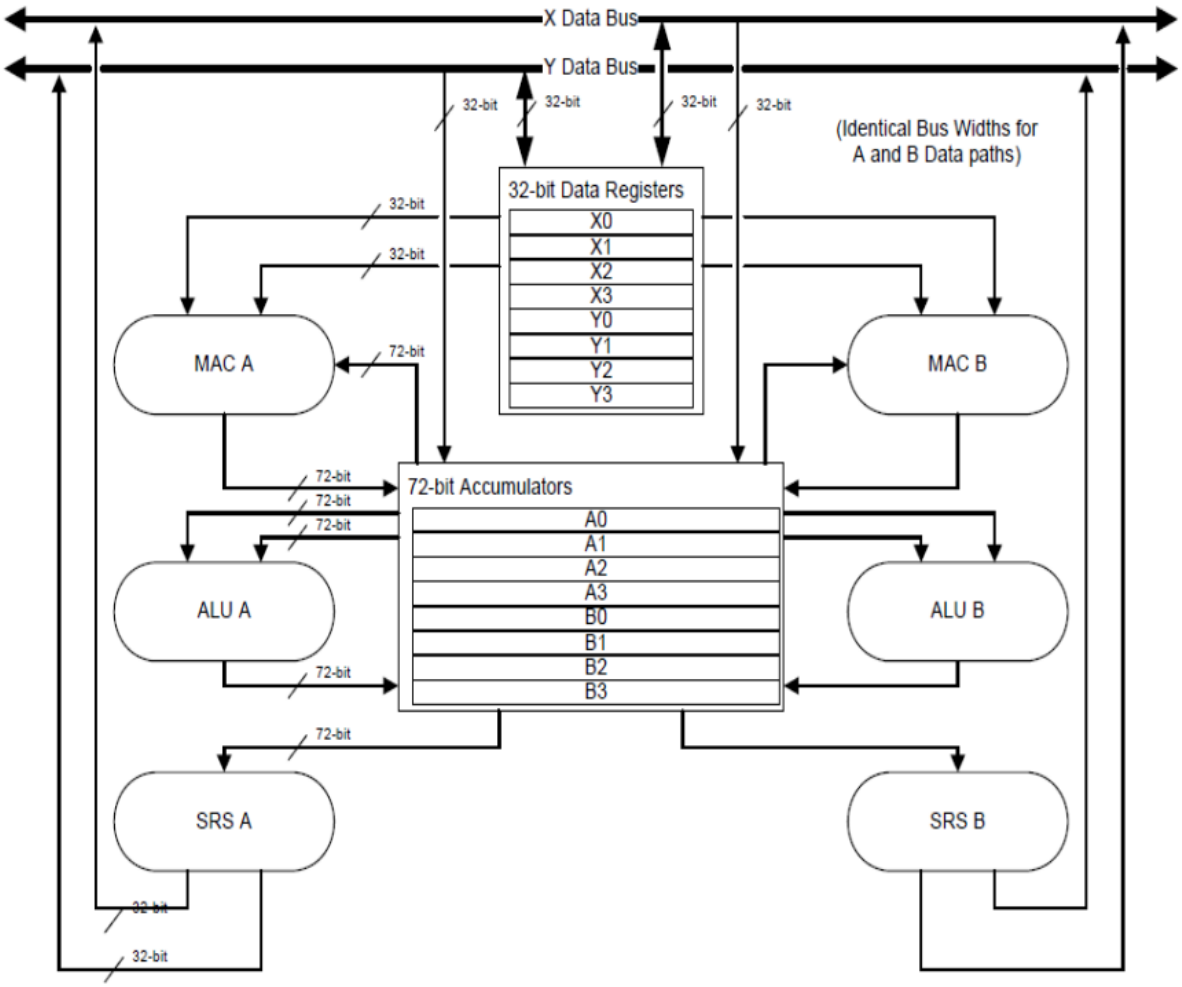
ДСП CS49844 процесор садржи четири 32-битна ДСП језгра који због рада са аритметиком непокретног зареза може да изврши две *MAC* (енг. *Multiply and Accumulate*) операције по такту (Слика 3.1).

Четири 32-битна ДСП језгра заснивају се на унапређеној Харвард архитектури, што значи да поседују посебну меморију за податке и посебну меморију за инструкције [15]. Меморија за податке је додатно подељена у две посебне целине (Табела 3.1).

Тип меморије	Језгро А	Језгро Б	Језгро Ц	Језгро Д
X, Y, P	60k Word SRAM	60k Word SRAM	60k Word SRAM	60k Word SRAM

Табела 3.1 Расположива меморија процесора CS49844

Свако језгро се састоји из јединице за контролу тока програма, два паралелна тока података (A и B) и паралелних јединица за генерисање адреса (енг. AGU – *Address Generation Unit*) која располаже са дванаест индексних регистара (i0-i11) и дванаест одговарајућих модуло регистара (mn0-mn11) који омогућавају различите адресне режиме [16]. Језгро садржи осам 72-битних акумулаторских регистара (a0-a3, b0-b3) намењених за чување крајњих и међу резултата MAC инструкције и других аритметичких операција. Акумулатори се састоје од три надовезана регистра означених као Guard (8b), High (32b), Low (32b), при чему се сваком делу може приступити засебно. Поред акумулаторских, у језгру се још налази и осам 32-битних регистара опште намене (x0-x3, y0-y3). Сваки податак мора да прође кроз све јединице тока података: MAC, ALU (енг. *Arithmetic-Logic Unit – аритметичко-логичка јединица*), која је одговорна за све логичке операције извршене над акумулаторима и SRS (енг. *Shifter/Rounder/Saturator* – померање, заокруживање, засићење).



Слика 3.2 Блок дијаграм DSP језгра Crystal 32

3.2 Методологија развоја ДСП апликација

Процес развоја софтверских апликација представља структуирани скуп активности који за циљ има развој софтверског производа. Софтверски производ се састоји од развијених програма и документације. Документација подразумева списак захтева које софтверска апликација треба да испуни, опис дизајна апликације и корисничка упутства. Овај процес непосредно утиче на квалитет и цену крајњег производа. Под ефикасним развојем софтвера подразумевамо да се са коначно дефинисаним обимом ресурса, у предвиђеном року, постигне производ високог квалитета. При томе, критеријум квалитета који одређују добро развијен софтверски производ су:

- функционалност,
- поузданост,
- употребљивост,
- ефикасност,
- одржавање,

- покретљивост.

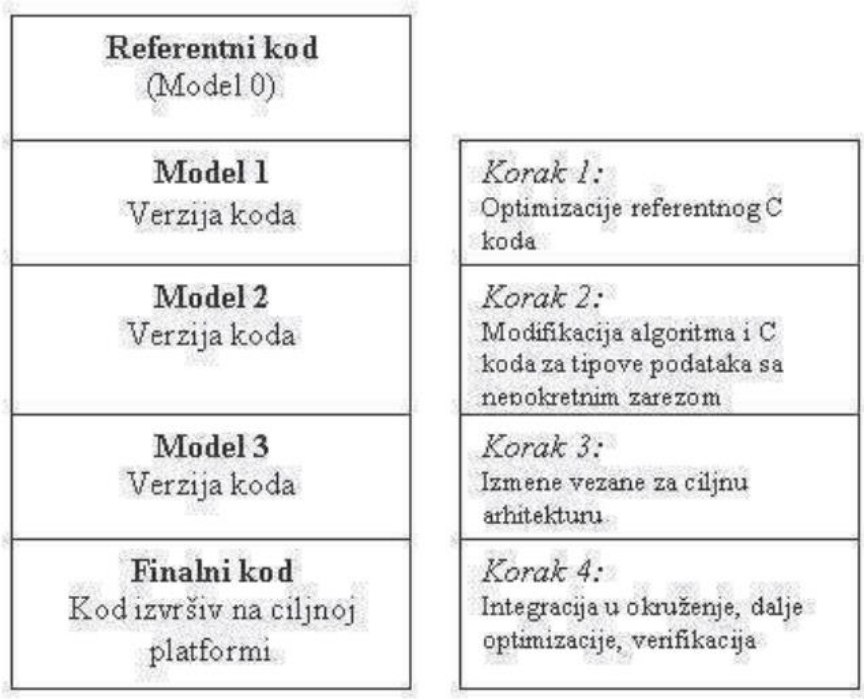
Уопштено, модели развоја софтвера су апстракције које помажу у процесу развоја софтвера. Модел представља формалан опис фаза у развоју софтвера и поступака карактеристичних за сваку од фаза.

Процес развоја софтверских апликација за системе засноване на ДСП процесорима садржи одређене специфичности у односу на процесе развоја на општенаменским платформама. Физичка архитектура циљног процесора значајно дефинише критичне захтеве: меморијска ограничења, тачност излазних одбирака, дужину програма и време извршења. Ови захтеви нису независни, па се током развоја мора водити рачуна о свим ресурсима платформе. На пример, побољшање тачности при обради одбирака доводи до већег броја инструкција, или смањење времена извршења може да се изведе на рачун повећања меморијских захтева.

Методологија за развој апликација на оваквим системима је примарно развијена за процесоре са аритметиком у непокретном зарезу и ослања се на C преводилац за ДСП, али је великим делом примењива и када се приступа развоју заснованом искључиво на асемблерском језику. Прописани су кораци 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. Модел 3 представља потпуно преводив C код за наменску платформу. Након успешно формираног Модела 3 следи детаљна процена утрошка ресурса (профилисање). Уколико су резултати профилисања задовољавајући, следи повезивање кода са програмским окружењем циљне платформе. Модел 0, Модел 1 и Модел 2 се преводе општенаменским преводиоцем, док се Модел 3 преводи преводиоцем за циљну платформу (Слика 3.3).

Треба напоменути да развој апликације за циљну платформу представља итеративни поступак. Ако се на основу процене утрошка ресурса закључи да решење не задовољава унапред познате критеријуме, приступа се модификовању решења полазећи од Модела 1 [18].



Слика 3.3 Ток имплементације софтвера на ДСП процесорима са аритметиком непокретног зареза

3.2.1 Модел 0 - референтни код

У оквиру изложене методологије подразумевано је да је почетна тачка развоја референтни алгоритам реализован као C програмски код. Ово је и најчешћа реализација алгоритма, јер је много лакше и брже написати алгоритамски код када се не води рачуна о ограничењима физичке платформе. Ако је алгоритам дефинисан у некој другој форми (нпр. у облику стандарда, неког другог програмског језика или математичког модела), ова методологија захтева да се прво направи C референтни код, који се потом може прилагодити различитим ДСП платформама. Намена референтног кода је испитивање и верификација алгоритма, тако да се код писања референтног кода води рачуна искључиво о исправности решења, квалитету и тачности излазних вредности, док се проблеми који се тичу имплементације занемарују. Резултат Модела 0 служи за испитивање сваке следеће измене у коду. Поређењем референтних резултата са резултатима модификованог кода можемо проверити да ли измена довело до нарушавања семантике кода, па самим тим уочити и открити грешке у ранојој фази развоја софтвера. Излаз сваке наредне фазе се сматра исправним ако одговара референтним вредностима Модела 0. Модел 0 се не сме мењати у даљем току израде апликације.

3.2.2 Модел 1 - функционална оптимизација С кода

Функционална оптимизација С кода прилагођена ДСП процесору подразумева измене у референтном коду у складу са хардверским проширењима процесора: броју регистра, величини меморије, итд. Ове измене се своде на организацију кода и података, тако да након превођења утрошак процесорских ресурса буде оптималан и у складу са захтевима. Неке од измена су замена индексног адресирања низова адресирањем преко показивача, смањење броја аргумената у функцији и прилагођавање петљи касније коришћеним хардверским петљама. Треба нагласити да су функционалне модификације, у овој фази развоја, такве да се Модел 1 преводи општенаменским преводиоцем. Након примене оптимизационих техника резултат извршавања Модела 1 мора бити идентичан на нивоу бита резултату извршавања Модела 0, затим се може прећи у следећу фазу развоја.

3.2.3 Модел 2 - прилагођење кода аритметици ДСП-а

Референтни код алгоритма је најчешће заснован на аритметици са покретним зарезом, која се без модификације не може превести преводиоцем за процесоре са аритметиком у непокретном зарезу. У овој фази развоја потребно је функционално оптимизован код прилагодити аритметици циљне платформе, али тако да је и даље преводив општенаменским преводиоцем. Због тога се често уводе класе које служе за емулацију типова у непокретном зарезу. На овај начин се обезбеђује код чија аритметика у потпуности прати аритметику циљног процесора, а и даље се може преводити и контролисано извршавати истим преводиоцем као и Модел 0 и Модел 1.

У оквиру скупа алата за развој софтвера за платформу CS498xx дате су C++ класе које служе за емулацију типова у непокретном зарезу: *fract* и *accum*. Прецизност ових типова одговара прецизности типова у непокретном зарезу код CCC2 преводиоца.

Највећи проблем приликом формирања Модела 2 је последица различите прецизности код аритметике са непокретним зарезом у односу на аритметику са покретним зарезом.

3.2.4 Модел 3 - превођење кода компајлером за циљну архитектуру

У овој фази развоја кода уводи се наменски ДСП преводилац, па модификације подразумевају прилагођење кода специфичностима С преводиоца за циљни процесор. Прво се направи симулаторски пројекат користећи CLIDE развојно окружење, па се у њега додају датотеке са изворним кодом Модела 2. Део кода задужен за читање и писање вредности у датотеку потребно је изменити тако да се користе функције које су део

стандардне библиотеке за циљну платформу. Затим се приступа отклањању грешака које пријављује CCC2 преводилац. То су најчешће грешке због разлике у подржаним C стандардима између CCC2 и општенаменског преводиоца. Свим глобалним променљивама потребно је придружити квалификатор меморијске зоне, а уколико се зона јасно не дефинише, сматра се да променљива припада X меморијској зони.

Након што је код успешно преведен и даје исправне резултате када се извршава на ДСП симулатору, приступа се процени утрошка ресурса – мерењу утрошка програмских и меморијских ресурса и оптимизацији кода. Први корак је примена техника оптимизације на C код, а након тога се делови кода који су и даље неефикасни, троше највише ресурса, оптимизују на нивоу асемблерског кода.

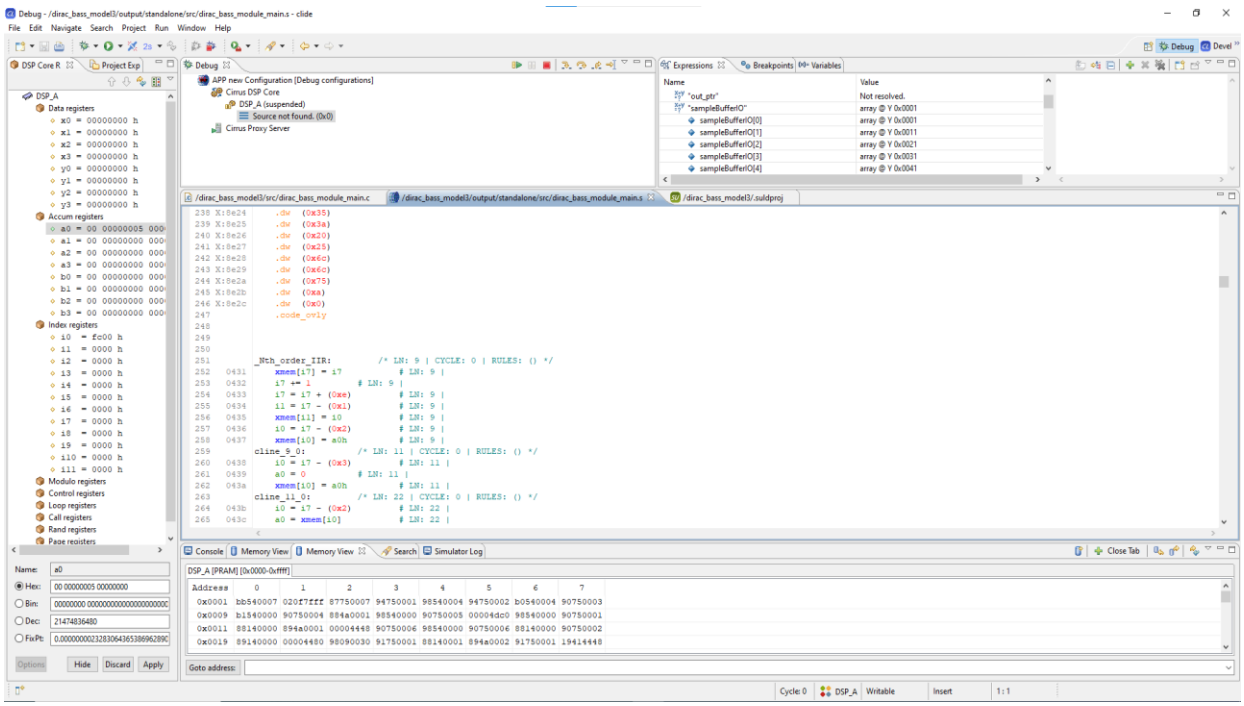
3.2.5 Финални модел – код извршив на циљној платформи

У оквиру Модела 3 се добија исправан и функционалан алгоритам, али сам код не може бити пуштен преко ДСП платформе. Потребно је још уградити неколико елемената, који зависе од платформе и који рукују неопходним ресурсима и пружају окружење за пуштање на платформи. За једну платформу је те елементе потребно направити једном и могуће их је поново касније користити. У финалном моделу се такође могу урадити још неке оптимизације везане за ограничења ресурса. Када се завршни код успешно спусти на ДСП платформу, врши се процес верификације.

3.3 Развојно окружење

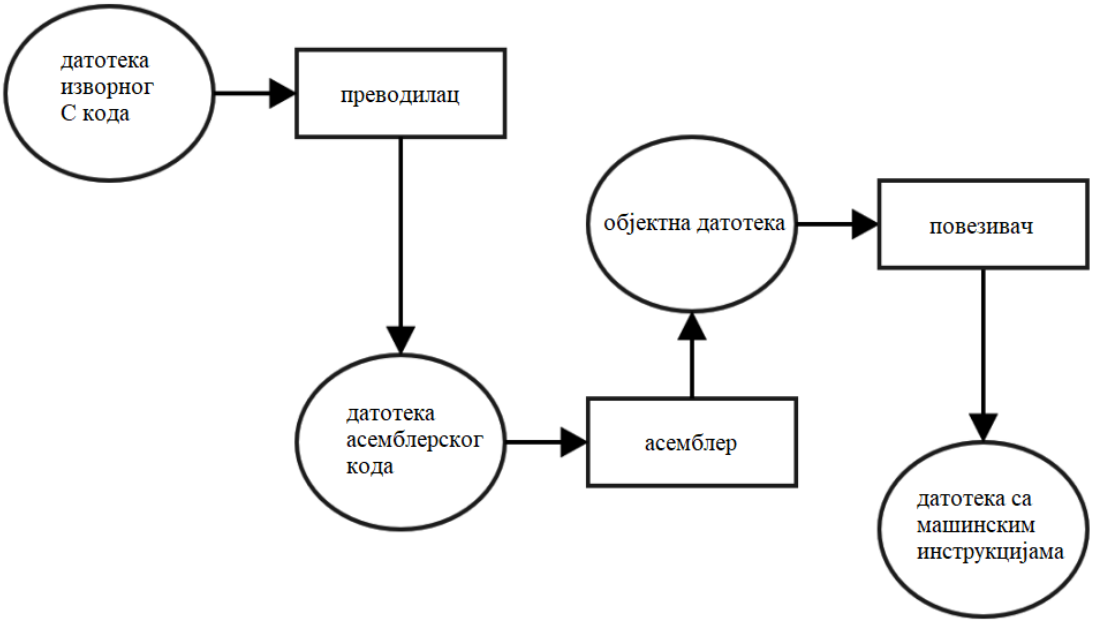
Модел 0, Модел 1 и Модел 2 развијани су у Microsoft Visual Studio окружењу које омогућава контролисано извршавање програма и рад са општенаменским преводиоцем.

За развој C кода током израде Модела 3, коришћено је интегрисано развојно окружење намењено дигиталним сигнал процесорима произвођача Cirrus Logic, CLIDE (енг. *Cirrus Logic Integrated Development Enviroment*) уз језичка проширења која нуде Cirrus Logic процесор и CCC2 преводилац (енг. *Cirrus Logic C Compiler*). Засновано је на Eclipse развојном окружењу које је широко распрострањено као основа многих развојних окружења због свог отвореног кода и мноштва корисних алата за развој програмске подршке. Развој програмске подршке за ДСП платформе у великој мери се олакшава употребом CLIDE-а због тога што таква окружења омогућују контролисано извршавање програма уз опције као што су преглед меморије и стања регистара као и могућност ефикасне детекције и отклањања грешака. Важна карактеристика овог окружења јесте да даје могућност превођења апликације и подршку за смештање и спуштање ДСП апликације на развојну плочу (Слика 3.4).



Слика 3.4 Контролисано извршавање програма у CLIDE окружењу

3.3.1 Програмски преводилац CCC2



Слика 3.5 Дијаграм тока превођења изворног кода

Модел 3 у овом задатку рађен је искључиво помоћу језичких проширења С језика за ДСП апликације и CCC2 преводиоца.

Програмски језик С не подржава аритметику у непокретном зарезу и меморијске зоне, због чега се уводи проширење С стандарда за наменске процесоре. CCC2, развијен у целости на Институту за рачунарску технику и рачунарске комуникације у Новом Саду, подржава овај стандард. Ток превођења код CCC2 преводиоца састоји се из неколико

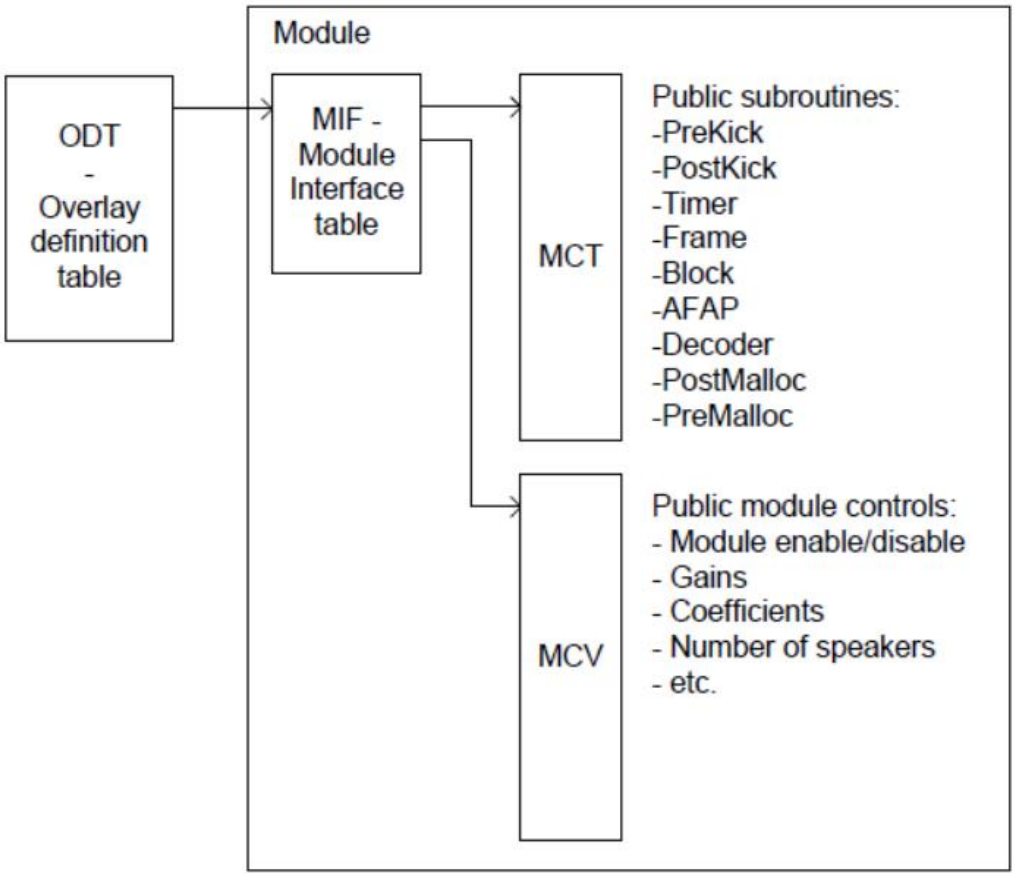
корака (Слика 3.5). CCC2 преводи C код и као резултат превођења даје датотеке са асемблерским кодом за циљну архитектуру. Након тога преводилац позива асемблер који на основу асемблерског кода генерише објектне датотеке. Последњи корак јесте позивање повезивача који као излаз даје софтверску библиотеку или извршну датотеку. Извршна датотека се уписује у меморију ДСП уређаја након чега следи њено извршавање.

3.4 Програмско окружење Cirrus Logic Framework

Cirrus Logic Framework (радни оквир) програмско окружење представља програмску подршку процесора која скраћује време и уложени рад за развој апликације уводећи неке од идеја и методологија из објектно орјентисаног програмирања. Језгро програмског окружења се састоји од једноставног оперативног система (енг. OS – *Operating System*) који ради као распоређивач за одређен број процесних ентитета. ОС представља мониторску петљу која позива рутине одговарајућих модула по унапред дефинисаном редоследу.

Сваки модул има свој јединствени спрежни подсистем (енг. MIF – *Module Interface*) којим је модул повезан са ОС-ом. Њега чини MIF табела која садржи показиваче на табеле са осталим спрежним информацијама. Табеле од највећег значаја су MCT (енг. *Module Call Table*) табела и MCV (енг. *Module Control Vector*) табела. ОС је повезан са модулима преко ODT (енг. *Overlay Definition Table*) табеле која садржи показиваче на MIF табеле свих учитаних модула [17].

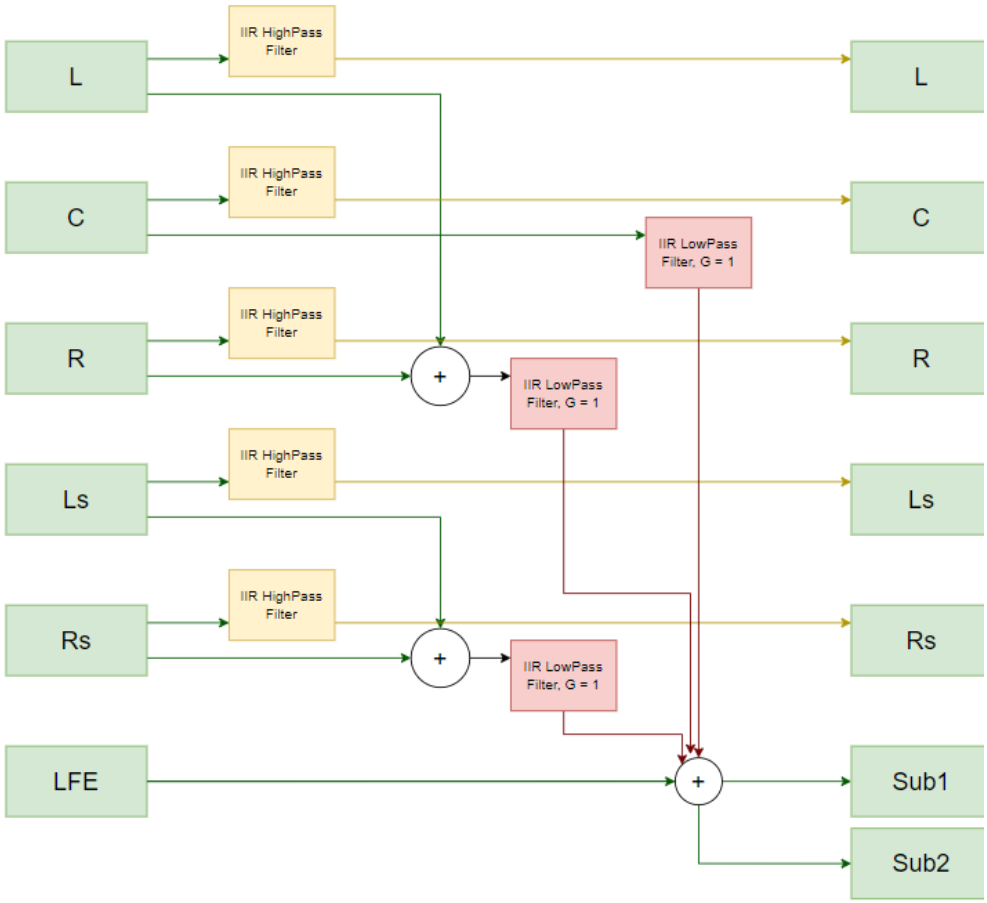
У зависности од конкретне табеле разликује се и садржај. MCV табела садржи низ јавно доступних конфигурационих параметара датог модула и тако омогућава конфигурисање модула од стране главног контролера уређаја. Нема унапред дефинисану структуру, већ програмер формулише њен садржај и структуру. За разлику од MCV табеле, MCT табела је унапред одређена и састављена од показивача на јавне рутине. Редослед елемената у табели је унапред дефинисан, а уколико дође до ситуације да рутина није дефинисана, на место њеног показивача се налази нула. Ове рутине ОС позива као одговор на појаву одговарајућих догађаја у систему и то су једине рутине модула којима ОС може приступити. Свака од њих има предефинисану намену и њиховим позивањем од стране ОС-а, модул одговара на одређене догађаје у систему (Слика 3.6).



Слика 3.6 Спрега модула са ОС-ом

4. Програмско решење

Алгоритам за управљање нискофреквентним садржајем генерише аудио токове од једноставних стерео до сложених аудио система формата 5.1 на улазу и 5.2 на излазу. Састоји се из елемената обраде филтрирања каскадама филтара са бесконачним импулсним одзивом (енг. *IIR – Infinite Impulse Response*) и елемената обраде кашњења са променом интензитета сигнала (енг. *Delay and Gain*) на основу којих се постепено гради целина која испуњава задатак управљања ниским фреквенцијама (Слика 4.1).



Слика 4.1 Блок шема алгоритма за управљање нискофреквентним садржајем

Први корак имплементације алгоритма је ремапирање улазног распореда канала специфичног за циљну платформу на одговарајући редослед канала потребан приликом вршења обраде над улазним сигналом. Након тога је потребно иницијализовати вредности елемената обраде за сваку фазу обраде, који се затим позивају као параметри функција. Затим следи вршење саме обраде сигнала по етапама (филтрирање или кашњење са променом интензитета). Крајњи задатак јесте вратити канале у правилан и очекивани распоред на излазу кроз поновно ремапирање канала.

4.1 Оптимизације током израде Модела 3

Током израде Модела 3 искоришћене су одређене технике оптимизације C кода за превођење CCC2 преводиоцем.

4.1.1 Оптимизација програмских петљи

Предност хардверских петљи јесте уштеда процесорских циклуса који се троше на инструкције за увећање индекса, проверу услова и инструкције скока. Да би CCC2 преводилац генерисао код који се ослања на хардверску петљу, потребно је да петља буде написана у форми *for* петље која испуњава следеће услове:

1. услов краја петље мора бити присутан,
2. све вредности у оквиру услова, изузев индекса, морају бити константе и
3. модификација индекса мора бити константа.

Унутар петље у функцији за ремапирање канала на улазу, током проласка кроз све улазе, уколико дође до замене централног и десног канала бројач петље треба два пута да се увећа јер се замена извршава у истом пролазу петље. Како би CCC2 преводилац генерисао хардверску петљу и тиме се смањило непотребно трошење ресурса, уведена је променљива *flag* која се поставља на 1 када дође до поменуте ситуације што не подразумева измену вредности бројача на одређену вредност унутар саме петље већ је увећава у складу са бројем пролаза кроз петљу. На овај начин, наменски преводилац може да искористи предности хардверске петље, чиме се утрошени циклуси при обради знатно смањују.

4.1.2 Оптимизација условних исказа

Условни искази који садрже поређење две променљиве, преводилац готово увек преводи у инструкцију одузимања, користећи аритметичко логичку јединицу и инструкцију гранања на основу регистра стања. Неефикасност приликом превођења потиче од тога што приликом извршења операције одузимања може доћи до прекорачења опсега типа променљивих које се пореде. Управо из тог разлога, израз у условном исказу

замењен је једном помоћном променљивом у коју је претходно смештен израз који смо желели да упоредимо.

4.1.3 Оптимизација приступа меморији

Поред оптимизације потрошње програмске меморије, ова оптимизациона техника доприноси и побољшању брзине извршавања кода. Истовремени приступ подацима у две различите меморијске зоне са аспекта CCC2 преводиоца омогућава се увођењем квалификатора меморијских зона који доприносе прецизном заузимању ресурса обе меморијске зоне. Овакав приступ омогућава преводиоцу да генерише паралелно читање или упис података, чиме се смањује коришћење програмске меморије и извршава се мање циклуса приликом извршавања кода.

4.1.4 Коришћење кружног бафера

Коришћењем кружног бафера прате се уписана и закаснела вредност одбирка. За имплементацију кружног бафера искористило се модуло адресирање које омогућава компајлерско проширење *CIRC_INC*. Модуло адресирање се постиже додавањем директиве за поравнање приликом заузимања меморијског низа. Приликом инкрементирања индексног регистра са одговарајућим nm регистром, вредност адресе у индексном регистру ће се вратити на почетак бафера када буде достигнута крајња адреса бафера. Уместо сталне провере да ли су показивачи за читање и упис стигли до краја бафера и у зависности од ситуације враћани на почетак или само прелазили на следећи елемент низа, ова измена у Моделу 3 је доста утицала на смањење утрошка ресурса.

4.2 Специфичности рада са компајлером CCC2

Уколико оптимизован код није написан специфичном синтаксом коју преводилац може да препозна и оптимизује за ДСП, као резултат произвешће се субоптимално решење које троши много више ресурса него што би требало.

Приликом израде Модела 3, код је у односу на Модел 2 морао доста да се модификује да би се смањио број извршених инструкција и меморије због компајлерског превођења инструкције на много сложенији начин него што их је могуће написати у асемблерском коду. Проблеми који су се у том смислу јављали јесу стално преузимање података са стека уместо да се памте у неком регистру, што је решено увођењем помоћне променљиве која је памтила тренутни податак. Затим, за промену бројача *for* петље унутар саме петље преводилац није пронашао могућност да искористити хардверску петљу уз неке додатне услове, што је даље условило додатним модификацијама кода у

виду увођења помоћне промењљиве на основу које би бројач могао адекватно да се увећа на жељени индекс који није унутар саме петље.

4.3 Функција remap_input

- void remap_inputs();

Функција remap_inputs() задужена је за почетни корак промене распореда канала. Унутар ње се на основу маске прерачунавају постојећи канали на улазу и по потреби им се мењају места ради лакшег каснијег обрађивања. Маска која у бинарном запису одговара активним каналима на улазу, прослеђује се као аргумент командне линије приликом превођења. Према захтевима алгоритма, канали иду редом: леви и десни предњи, централни, леви и десни просторни и на самом крају LFE канал уколико постоји. Оваквим редоследом канала олакшан је начин обраде парова канала (леви и десни предњи, леви и десни просторни), као и посебан приступ LFE каналу који је из тог разлога смештен на сам крај.

4.4 Функција module_init

- void module_init();

Структуре које чине елементи обраде се иницијализују на одговарајуће вредности за сваку фазу обраде независно и додељују им се послати коефицијенти, начин обраде: IIR филтар N-тог реда или кашњење уз задато појачање/смањење и у зависности од изабраног начина обраде додатни параметри. Додела вредности елементима обраде врши се у функцији module_init(). Иницијализоване структуре садрже све потребне информације за сваку од обрада и из тог разлога се прослеђују као параметар функције обраде описане у наставку. Укупан број улазних канала, број сабирница и број излаза се такође прерачунава у овој функцији. Сабирнице (магистрале) у блок шеми представљају излаз из нископрпопусног филтра који садржи збир улазних канала пуног опсега и улазног LFE канала.

4.5 Функција processing

- void processing();

Processing() функција садржи позиве функција потребних обрада, који уз одређени редослед прате секције обраде.

4.5.1 Функција processing_iir

- void processing_iir (IIR_cell* cell, double [][BLOCK_SIZE], double [][BLOCK_SIZE] , int cells_num);

Processing_iir() функција служи за обраду звука применом IIR филтра N-тог реда. Ова функција пролази кроз низ елемената обраде за задату IIR секцију и позива функцију *Nth_order_IIR()* са одговарајућим параметрима.

- double Nth_order_IIR (IIR_cell* iir_cell, double input);

У овој функцији врши се филтрирање улазног одбирка улазног сигнала, коришћењем низа (каскаде) биквад (енг. *biquad*) филтара. Колико филтара у низу ће бити присутно, зависи од тога да ли се користи нископропусни филтар, високопропусни филтар или свепропусни филтар, што је дато у структури сваког елемента обраде.

4.5.2 Функција processing_delay_gain

- void processing_delay (delay_gain* delay_gain_cell, double [][BLOCK_SIZE], double [][BLOCK_SIZE]);

Processing_delay обрађује кашњење сигнала уз задато појачање.

4.6 Функција remap_outputs

- void remap_outputs (double out_buff[][BLOCK_SIZE]);

Када су сви канали прошли кроз све фазе обраде, потребно их је вратити у првобитан распоред, што се врши позивом функције remap_outputs() која као параметар прослеђује бафер у којем се налазе тренутни излази. Провером излазне маске која је такође дата као аргумент командне линије, а која представља активне канале на излазу у бинарном формату, канали се постављају у редослед: леви предњи, централни, десни предњи, леви просторни, десни просторни и сабвуфер излази на 7. и 14. индекс улазно/излазних бафера, по унапред утврђеном распореду излаза на платформи.

5. Испитивање и резултати

На крају сваког модела је неопходно проверити исправност решења, тако што се излазни вектори модела пореде са излазним векторима Модела 0. Скуп улазних вектора је веома често део референтног софтверског пакета (Модела 0). Ови вектори се формирају тако да се испита свака грана обраде, па је важно испитивање извршити над целим скупом улазних вектора. Поређење излазних резултата модификованог кода са референтним, може да се изведе на неколико начина: слушним тестовима, поређењем датотека на нивоу бита и анализом спектра излазних сигнала.

Улазни канал	**Фреквенција синусног тона**
L	500Hz
C	800Hz
R	1500Hz
Ls	2000Hz
Rs	3000Hz
LFE	50Hz

Табела 5.1 Садржај тестног вектора input_0x00009f.wav

Како не постоје унапред задати тестни вектори, помоћу којих би се могла испитати исправност алгоритма, помоћу скрипте писане у python програмском језику, генерисан је већи број тест вектора који су се међусобно разликовали по броју канала. Тестови су формирани тако да садрже и ниске и високе фреквенције како би се испитивање могло правилно извршити и тако утврдити исправност алгоритма (Табела 5.1). Утврђено је да

имплементација исправно ради тако што је спектралном анализом потврђено да има одговарајући мултитон сигнал наспрам шеме обраде (Слика 4.1).

Главни начин поређења који је коришћен за пројектни рад јесте поређење на нивоу бита које се врши тако што се испитна и референтна датотека пореде на нивоу бинарног записа. Идентичност резултата очекује се између Модела 0 и Модела 1, и између Модела 2 и Модела 3. Промена аритметике приликом преласка из Модела 1 у Модел 2 доводи до промене прецизности (нпр. као последица заокруживања). У складу са алгоритмом који се имплементирао, уведене су одређене толеранције грешке јер је закључено да оне не утичу значајно на крајњи резултат рада система.

Један од начина поређења датотека на нивоу бита је алат PCMCompare, који се позива из командне линије, са следећим параметрима:

- `PCMCompare.exe -b32  <датотека1> <датотека2>`

где додатна опција „-b32” означава број бита по одбирку (у нашем случају 32b)

Резултат поређења дат је у форми табеле. Уколико је једна датотека краћа од друге, поређење ће се прекинути када се стигне до краја краће датотеке и о томе ће бити исписано обавештење. У табели се налазе редом: број бита разлике, број одбирака који садрже грешку, проценат одбирака у односу на све одбирке у датотекама и адресу прве појаве грешке. На дну табеле налази се укупан број одбирака са грешком.

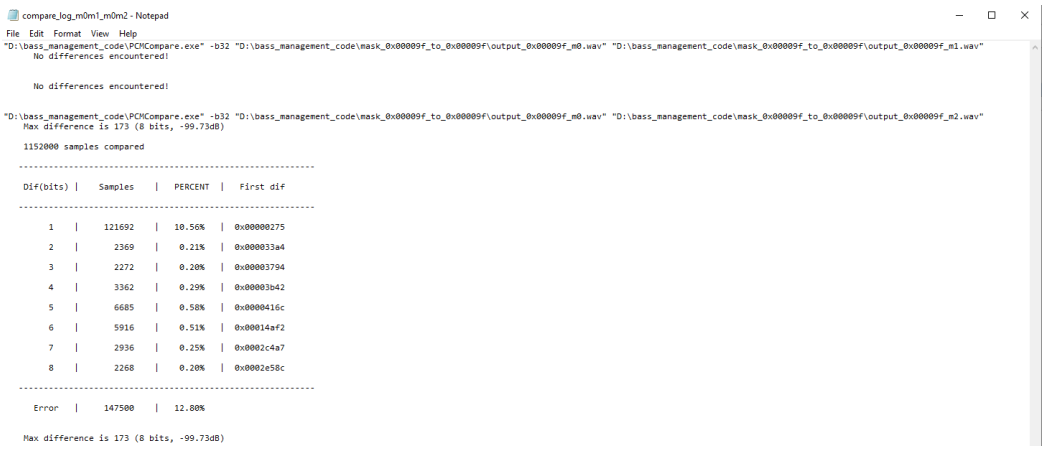
5.1 Аутоматизација процеса испитивања

Испитивање апликације, тако што се секвенцијално извршава велики број испитних вектора, дуготрајан је посао. Такође, након сваке измене у коду, исправке грешке или оптимизације, неопходно је поновити испитивање над целим скупом улазних вектора. Овај процес је веома спор уколико се сваки тест конфигурише, пушта и проверава ручно, због чега се тежи аутоматизацији процеса испитивања. За аутоматизацију испитивања најчешће се користе скрипт језици: за покретање фаза испитивања, подешавање параметара и улазних вектора, као и за запис резултата у одговарајућу датотеку која представља извештај о успеху испитивања.

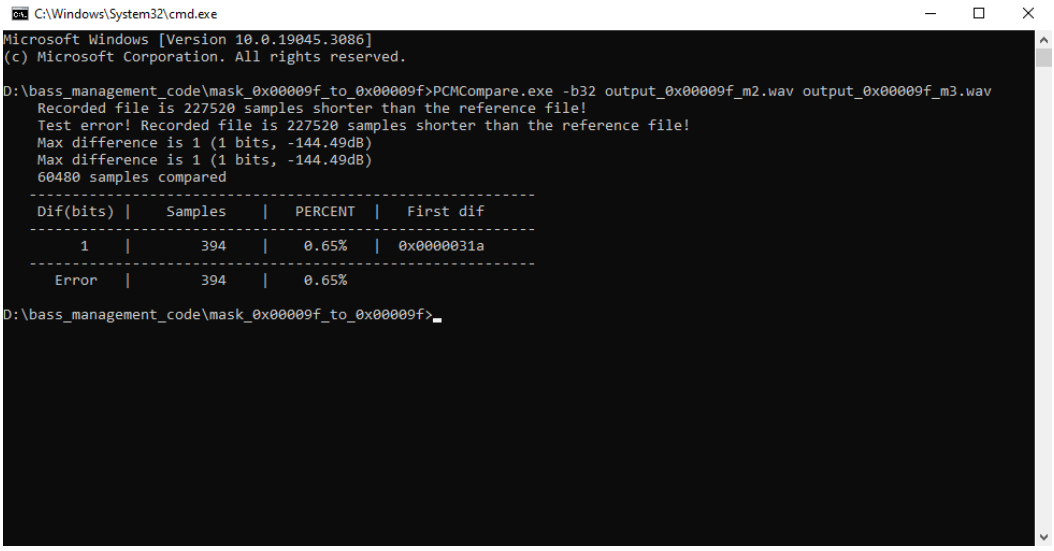
Први корак аутоматизованог испитивања јесте прављење комбинација улазне и излазне маске које одређују број канала, на основу којих се даље врши формирање тест директоријума за сваку од комбинација. Затим се генеришу улазне .wav датотеке и смештају у одговарајуће директоријуме спрам улазне маске. За добијене комбинације покреће се извршавање редом Модела0, Модела1 и Модела2 који као аргументе командне линије прослеђују путању до улазне .wav датотеке, која је претходно генерисана, путању до излазне .wav датотеке, која треба да се генерише, као и

одговарајућу улазну и излазну маску спрам директоријума у којем се налазе датотеке (Слика 5.1). Путање до потребних датотека формиране су као низ стрингова који се спајају заједно и чине путању. Када се све фазе (Модел0, Модел1, Модел2) изврше, врши се поређење излазних вектора помоћу PCMCompare.exe извршне датотеке, а резултати поређења бележе се у текстуалним датотекама за евидентирање активности током рада апликације (енг. *logging*) *compare_log_m0m1_m0m2.txt* (Слика 5.2). Поређење између Модела 3 и Модела 2 рађено је ручно (Слика 5.3, Слика 5.4).

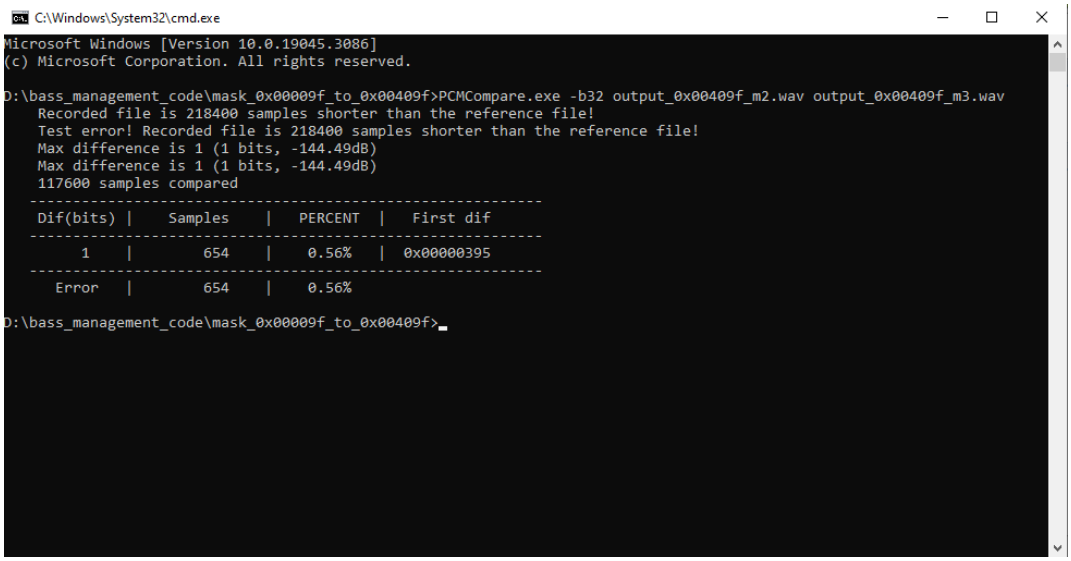
Слика 5.1 Испис аутоматизованог испитивања у конзоли python скрипте



Слика 5.2 Пример текстуалне датотеке након покретања скрипте



Слика 5.3 Пример ручног поређења излаза 5.1 аудио система



Слика 5.4 Пример ручног поређења излаза 5.2 аудио система

Због постојећих разлика између аритметика покретног и непокретног зареза, квантизације коефицијената филтара и форме имплементације филтра (коришћена је прва директна форма) одређени излази имају већу разлику на нивоу бита. Међутим, разлика није била већа од -96 dBFS (енг. *Decibels relative to full scale* – децибела пуне скале), што приликом слушних тестова не даје приметну разлику у односу на референтни излаз.

5.2 Утрошак ресурса у модулу за управљање нискофреквентним садржајем

Након завршетка имплементације алгоритма за управљање нискофреквентним садржајем, урађено је мерење потрошње ресурса (профилисање) које даје информације о утрошеним ресурсима, односно, говори да ли се програм може извршавати на циљној платформи, као и да ли број утрошених циклуса испуњава задата временска ограничења. Утрошак меморије је изражен кроз број заузетих речи, односно 32-битних меморијских локација. У радном окружењу је могуће добити само број утрошених циклуса, тако да се утрошак процесорског времена у MIPS-има (енг. *Million Instructions Per Second* – милион инструкција по секунди) рачуна по формули:

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

Упоредном анализом C решења и асемблерског решења, чији је развој текао паралелно, утврђено је да је C код потребно профилисати у складу са ограничењима асемблерске имплементације. Иницијално, имплементација алгоритма је ограничена за 11.2 аудио системе (Табела 5.2). Мерењем потрошње циклуса (Табела 5.3) за такав

систем, закључено је да не испуњава ограничења платформе. С обзиром да имплементација не задовољава услове ограничења платформе ни након оптимизације С кода (Табела 5.4), одлучено је да се имплементација сведе на 5.2 излазну конфигурацију звучника (Табела 5.6).

Мерење заузећа меморије врши се анализом генерисане .MAP или .lst датотеке након превођења. Датотека .MAP се након превођења налази унутар излазне датотеке и садржи информације о заузетој меморији од стране целе апликације, као и називе, адресе и иницијалне вредности свих симбола. Табела 5.7 приказује потрошњу меморије коришћене платформе, подељена на заузету програмску меморију три зоне: X меморија, Y меморија и P меморија која представља програмску меморију.

Број улазних канала пуног опсега (Nch_full)	11
Број LFE канала на улазу (Nch_LFE)	1
Укупан број канала на улазу	12
Број сабирница = Nch_full/2 + N_center + Nch_LFE	7
Величина блока обраде (BLOCK_SIZE)	16
Број малих или великих звучника на излазу	11
Број сабвуфера на излазу	2
Укупан број звучника на излазу	13
Фреквенција одабирања (Fs)	48000

Табела 5.2 Карактеристике имплементације за 11.2 аудио систем

Функција обраде	**Почетак (број циклуса)**	**Почетак (број циклуса)**	**Укупно циклуса**	**MIPS**
processing()	3412489	3811173	398684	1196.05
processing_iir_stage1	3412516	3534563	122047	366.141
processing_iir_stage2	3534595	3613144	78549	235.647
proceesing_delay_stage3	3613176	3683103	69927	209.781
processing_iir_stage4	3683135	3794736	111601	334.803
processing_delay_stage5	3794768	3811135	16367	49.101
Nth_order_IIR	1961156	1961540	384	1.152
remap_inputs()	3412516	3534563	122047	366.141
remap_outputs()	3534595	3613144	78549	235.647
Укупно (обрада):				**1797.84**
module_init()	3412489	3811173	398684	1196.05

Табела 5.3 Утрошак MIPS-а пре оптимизације кода за аудио систем 11.2

Функција обраде	Почетак (број циклуса)	Крај (број циклуса)	Укупно циклуса	MIPS
processing()	135405	342088	206683	619.677
processing_iir_stage1	135427	201248	65821	197.463
processing_iir_stage2	201268	243527	42259	126.777
proceesing_delay_stage3	243547	274743	31196	93.588
processing_iir_stage4	274763	334906	60143	180.429
processing_delay_stage5	334926	342066	7140	21.42
Nth_order_IIR	135508	135722	214	0.642
remap_inputs()	134391	135385	994	2.982
remap_outputs()	342088	343363	1275	3.825
Укупно (обрада):				**626.484**
module_init()	4636	129227	124591	373.773

Табела 5.4 Утрошак MIPS-а након оптимизације кода за аудио системе 11.2

Број улазних канала пуног опсега (Nch_full)	5
Број LFE канала на улазу (Nch_LFE)	1
Укупан број канала на улазу	6
Број сабирница = Nch_full/2 + N_center + Nch_LFE	4
Величина блока обраде (BLOCK_SIZE)	16
Број малих или великих звучника на излазу	5
Број сабвуфера на излазу	2
Укупан број звучника на излазу	7
Фреквенција одабирања (Fs)	48000

Табела 5.5 Карактеристике имплементације за 5.2 систем

Функција обраде	Почетак (број циклуса)	Крај (број циклуса)	Укупно циклуса	MIPS
processing()	53506	150275	96769	289.935
processing_iir_stage1	53528	85110	31582	94.746
processing_iir_stage2	85130	106890	21760	65.28
proceesing_delay_stage3	106910	116725	9815	29.445
processing_iir_stage4	116745	146696	29951	89.853
processing_delay_stage5	146716	150253	3537	10.611
Nth_order_IIR	135508	135722	214	0.642
remap_inputs()	52756	53486	730	2.19
remap_outputs()	150295	151287	992	2.976
Укупно (обрада):				**295.101**
module_init()	3385	48808	45423	136.269

Табела 5.6 Утрошак MIPS-а након оптимизације кода за аудио систем 5.2

X меморија	Y меморија	L меморија	P меморија
12532	2475	-	1608

Табела 5.7 Потрошња меморије (број речи) за 5.2 систем

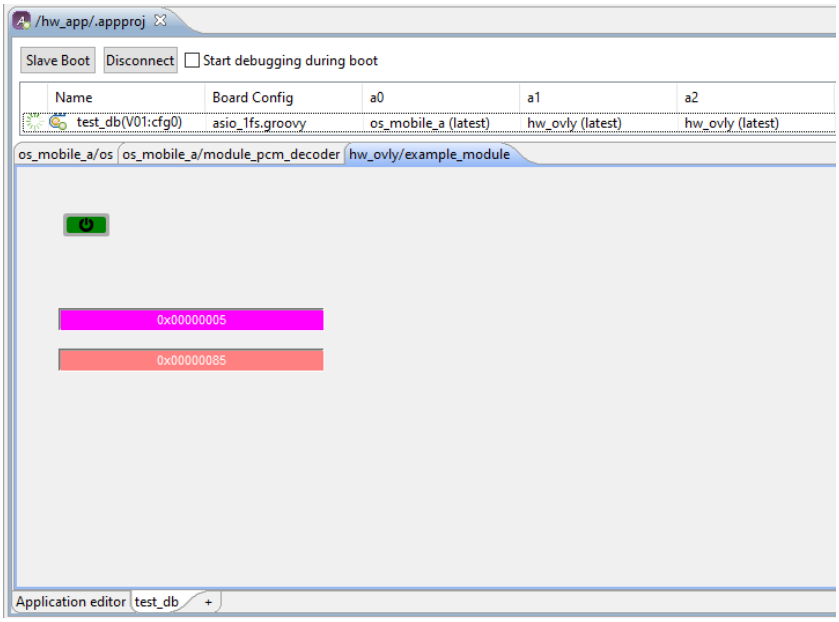
5.3 Верификација рада на дигиталном сигнал процесору

Како у развојном окружењу CLIDE постоји подршка за процесор CS48L20 у виду аутоматског генерисања хардверског пројекта и ДСП апликације, рад модула је помоћу слушних тестова додатно потврђен извршавањем и на CS48L20 чипу. Већ имплементирано решење може да се користи и на овом чипу зато што су засновани на истој архитектури ДСП језгра, при чему су разлике само у броју језгара и величини меморије.

CS48L20 процесор се састоји од два језгра (А и Б) са две одвојене меморије за податке, X и Y, и програмске меморије. Унутар сваког језгра величина меморије износи 15k.

5.3.1 Слушно тестирање

Извршавање је прилагођено ограничењима коришћене платформе, тако да је искоришћена стерео конфигурација. CLIDE окружење омогућава извршење кода директно на CDB49x развојној плочи из развојног окружења, чиме скрива од корисника, који развија софтвер, детаље о процесу спуштања ДСП апликације на плочу и конфигурисања плоче. CLIDE окружење обезбеђује да се помоћу графичке спреге лако изабере конфигурациона датотека и покрене извршавање апликације на плочи (Слика 5.5).



Слика 5.5 Пример конфигурације модула и покретања ДСП апликације

У складу са начином тестирања изабрана је *asio_1fs.groovy* конфигурација, која конфигурише плочу да аудио улаз прими са звучне картице помоћу I^2S спреге. Улазни тест вектор је пуштен на улаз процесора помоћу RT-AG звучне картице [19]. ОС, уз декодер PCM улазног тока и модул су распоређени на ДСП А језгро. Додатне конфигурације, у виду улазне и излазне маске, које одређују број канала, дају се као део конфигурације модула. На пример, маска 0x5 на маску 0x85 значи да ће се од стерео улаз (леви и десни канал) добити стерео излаз, где ће нискофреквентни садржај са та два канала бити прослеђен на посебан канал за излаз на сабвуфер звучник. Резултат слушног тестирања се може слушати преко аналогног излаза, или снимити помоћу RT-AG звучне картице (Слика 5.6).

Слика 5.6 Окружење коришћено током извођења слушних тестова

5.4 Поређење резултата са асемблерском имплементацијом модула

Истовремено са развојем C-заснованог решења овог модула (уз компајлерска проширења за циљну платформу), урађена је имплементација овог модула директно у асемблерском језику, без коришћења референтног C кода, само на основу блок шеме обраде.

У току развоја асемблерског решења, имплементирано решење у C-у је коришћено као додатна провера. Модели 0, 1 и 2 су коришћени током развоја за утврђивање разлога неисправног рада асемблерског решења:

- разлике које настају при прелазу са аритметике у покретном на аритметику у непокретном зарезу,
- прекорачење опсега приликом акумулације бројева у аритметици непокретног зареза,
- одређивање погодног формата за константе и коефицијенте и
- поређење ради испитивања природе шума у излазном сигналу (нпр. шум квантизације).

На самом крају израде, урађено је поређење C и асемблерског решења. Иако је иницијално време за израду C решења (Модел 0, 1, и 2) краће од времена за израду асемблерског решења, израда Модела 3 захтева добро познавање рада преводиоца за циљну платформу, односно начин писања C кода како би омогућио преводиоцу да искористи доступне оптимизације. Упознавање са радом преводиоца и његовим специфичностима захтева доста више времена, него имплементација директно у асемблеру, где је могуће одмах искористити предности архитектуре процесора. Преводилац за C често генерише субоптимално решење са већим бројем инструкција у односу на асемблерско решење, што се одражава на утрошак програмских ресурса (циклуса), као и саме програмске меморије (Табела 5.8). Додатно, C преводилац може да генерише и локалне променљиве на стеку, док је у асемблеру ово могуће избећи коришћењем привремене меморије или избором неискоришћених регистара.

Решење	**X меморија**	**Y меморија**	**P меморија**	**MIPS**
C	12532	2475	1608	295.101
Асемблер	3046	1497	572	37.041

Табела 5.8 Поређење утрошка ресурса C и асемблерског решења за 5.1 конфигурацију

6. Закључак

У овом раду развијен је алгоритам за управљање нискофреквентним садржајем сложених аудио система 5.1 и извршена је његова имплементација и оптимизација. Пре почетка рада, било је потребно упознати се са циљном платформом на којој ће се алгоритам извршавати, као и проучити методологију развоја ДСП апликација кроз моделе и начине обраде сигнала.

Током израде пролазило се кроз неколико фаза. Након потребног упознавања са циљном платформом, различите начине обраде звука и формирања тестних случајева, осмишљен је одговарајући алгоритам што представља прву фазу.

У другој фази, имплементиран је референтни код алгоритма уз одговарајуће обраде у C програмском језику и развојном окружењу Microsoft Visual Studio. Када се на тај начин утврдила исправност алгоритма, настављен је даљи ток израде.

У наредним фазама имплементиране су оптимизоване верзије почетног алгоритма кроз Модел 0, Модел 1, Модел 2 и Модел 3. Када је проверена исправност свих излазних вектора у односу на почетни референтни излаз, вршено је мерење утрошка ресурса и додатна оптимизација кода да би се они налазили у границама платформе.

Алгоритам за управљање нискофреквентним садржајем проверен је поређењем на нивоу бита свих тестних излазних сигнала, која су прошла очекивано. Овом методом провере верификовано је да је излазни сигнал исправан и да се обрађени садржај финалног модела подудара до одређене границе са референтним излазом.

Иако утрошак ресурса омогућава ефикасно извршавање алгоритма на ДСП платформи за систем са конфигурацијом звучника 5.2, могуће је унапредити постојећу имплементацију како би алгоритам био употребљив и за сложеније аудио системе од система 5.2. То би се првенствено могло постићи додатним изменама у самој логици

алгоритма још у току израде референтног кода, а затим и додатним оптимизацијама у виду боље расподеле података по меморијским зонама како не би дошло до презасићености једне зоне, док је друга минимално искоришћена. Додатно, поређењем са имплементираним асемблерским решењем, утврђено је да је писањем директно у асемблерском језику, без ослањања на CCC2 преводилац, могуће направити решење које подржава већи број канала и звучника. Овим се добија већи степен оптимизације кода, што доприноси мањем утрошку ресурса и самим тим би примена алгоритма на системе сложеније од 5.2 постала могућа.

7. Литература

[1] „The evaluation of home theater”, S.Kindig, Crutchfield New Media, доступно на мрежи: https://www.crutchfield.com/S-BALeHIUAXBX/learn/home-theater-history.html [приступљено: 26.06.2023]

[2] “Immersive sound: the art and science of binaural and multi-channel audio”, A. Roginska, P. Geluso, Eds, First published. New York, London: Routledge, Taylor & Francis Group, 2018.

[3] „Преглед хардверске и софтверске архитектуре аудио система за дигиталну обраду сигнала“, Ј. Ковачевић, доступно на мрежи: https://www.rt-rk.uns.ac.rs/sites/default/files/materijali/predavanja/AADSP%20Audio%20sistemi%20i%20interfejsi.pdf [приступљено: 10.07.2023]

[4] „Smart Multi-Agent Framework for Automated Audio Testing“, J. Kovacevic, U. Radujko, M. Djukic, and T. Novkovic, ELEKTRON ELEKTROTECH, vol. 29, no. 1, pp. 59–68, Feb. 2023, doi: 10.5755/j02.eie.33222.

[5] „Home Theater Market by Product, Distribution Channel, and Geography - Forecast and Analysis 2023-2027“, доступно на мрежи: https://www.technavio.com/report/home-theater-market-industry-analysis [приступљено: 10.07.2023]

[6] „Једно решење софтверске архитектуре пријемника аудио и видео садржаја засновано на дигиталном сигнал процесору“, М. Антонић, Зборник радова Факултета техничких наука, Едиција: Техничке науке - зборници, Број 11/2018, Година XXXIII, Свеска 11, Факултет техничких наука Нови Сад, 2018.

[7] „Altec Lansing Audio / Video Equipment #ADA106“, Tradeloop Corporation, доступно на мрежи: https://www.tradeloop.com/p/audio-video-equipment/alteclansing/ada106?catalogid=668593 [приступљено: 10.07.2023]

[8] „Audio, Electronics, Music, Tips “6 Best Soundbars For Home Audio“, FromDev, 25 December 2015, доступно на мрежи: https://www.fromdev.com/2015/12/best-soundbars-home-audio.html [приступљено 10.07.2023]

[9] „Are Soundbars Worth It?“, G. Morrison, Forbes, 28 July 2015, доступно на мрежи: https://www.forbes.com/sites/geoffreymorrison/2013/07/09/are-soundbars-worth-it/#11b9587f4d25 [приступљено 10.07.2023]

[10] „Choosing the Best Soundbar“, Consumer Reports, 7 June 2018, доступно на мрежи: https://www.consumerreports.org/electronics-computers/sound-bars/buying-guide/ пристипљено [10.07.2023]

[11] 2.1.2ch Dolby Atmos Soundbar with Wireless Subwoofer, доступно на мрежи: https://www.eko-entertainment.com.au/ksb212wda-dolby-atmos-soundbar-wireless-sub [приступљено: 10.07.2023]

[12] „Bass Management and the LFE Channel”, A. Grimani, Sound&Vision, August, 2005, доступно на мрежи: https://www.soundandvision.com/content/bass-management-and-lfe-channel [приступљено: 10.07.2023]

[13] „What is the LFE channel?”, Dolby Laboratories, Inc 2000, доступно на мрежи: https://www.dolby.com/uploadedFiles/Assets/US/Doc/Professional/38_LFE.pdf [приступљено: 26.06.2023]

[14] „Dirac Live Bass Control”, F. Rosencratz, M.U.Andersson, L. Bränmark, доступно на мрежи: https://confluence.dirac.services/display/DLS/Dirac+Live+Bass+Control?preview=/142848300/142848301/Bass%2BControl%2Bin%2BLive.pdf [приступљено: 27.06.2023]

[15] „Архитектуре и алгоритми дигиталних сигнал процесора 1“, В. Ковачевић, М. Поповић, М. Темеринац, Н. Теслић, ФТН издаваштво, Нови Сад, 2005

[16] „CS498xx_Datasheet“, Cirrus Logic, Inc, 2013

[17] „Архитектура и алгоритми ДСП 2 - практикум за лабораторијске вежбе”, Нови Сад, 2013

[18] „A C compiler based methodology for implementing audio DSP applications on a class of embedded systems“, M. Djukic, N. Cetic, J. Kovacevic and M. Popovic, IEEE International Symposium on Consumer Electronics, Vilamoura, Portugal, 2008, pp. 1-4, doi: 10.1109/ISCE.2008.4559481

[19] „Tehničko rešenje softverske arhitekture digitalnog audio grebera i plejera RT-AG“, N. Pekez, J. Kovačević, M. Beljulji, International Conference for Electronics, Telecommunications, Computers, Automatic Control and Nuclear Engineering (ETRAN), Kladovo, Serbia, June, 2017.

[20] „Архитектуре и алгоритми дигиталних сигнал процесора“, др J. Ковачевић, Д. Бокан, збирка задатака и лабораторијски практикум