



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: Нивес Капроцки

Број индекса: РА139/2012

Тема рада: Израда блока аудио обраде за виртуализацију звучника на
ДСП платформи

Ментор рада: Доц. др Јелена Ковачевић

Нови Сад, јул, 2016.



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:	Нивес Капроцки		
Ментор, МН:	Доц. др Јелена Ковачевић		
Наслов рада, НР:	Израда блока аудио обраде за виртуализацију звучника на ДСП платформи		
Језик публикације, ЈП:	Српски / латиница		
Језик извода, ЈИ:	Српски		
Земља публиковања, ЗП:	Република Србија		
Уже географско подручје, УГП:	Војводина		
Година, ГО:	2016		
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)	7/24/6/5/13/0/0		
Научна област, НО:	Електротехника и рачунарство		
Научна дисциплина, НД:	Рачунарска техника		
Предметна одредница/Кључне речи, ПО:	Дигитална обрада сигнала, виртуализација звучника, системи окружења, саунд бар		
УДК			
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад		
Важна напомена, ВН:			
Извод, ИЗ:	У оквиру рада је имплементирано једно решење модула за виртуализацију саунд звучника на ДСП процесору фирме Cirrus Logic за саунд бар системе.		
Датум прихваташа теме, ДП:	5.07.2016.		
Датум одбране, ДО:	12.07.2016.		
Чланови комисије, КО:	Председник:	Доц. др Миодраг Ђукић	
	Члан:	Доц. др Иван Каштелан	Потпис ментора
	Члан, ментор:	Доц. др Јелена Ковачевић	



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:	Nives Kaprocki	
Mentor, MN:	PhD Jelena Kovačević	
Title, TI:	DSP implementation of speaker virtualizer	
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:	2016	
Publisher, PB:	Author's reprint	
Publication place, PP:	Novi Sad, Dositeja Obradovica sq. 6	
Physical description, PD: (chapters/pages/ref./tables/pictures/graphs/appendices)	7/24/6/5/13/0/0	
Scientific field, SF:	Electrical Engineering	
Scientific discipline, SD:	Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, S/KW:	Digital signal processing, speaker virtualization, surround systems, sound bar	
UC		
Holding data, HD:	The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, N:		
Abstract, AB:	This thesis describes one solution for an implementation of a surround speaker virtualization module for Cirrus Logic digital signal processor in sound-bar systems.	
Accepted by the Scientific Board on, ASB:	5.07.2016.	
Defended on, DE:	12.07.2016.	
Defended Board, DB:	President: Member: Member, Mentor:	PhD Miodrag Đukić PhD Ivan Kaštelan PhD Jelena Kovačević
		Menthor's sign

Zahvalnost

Zahvaljujem se svom mentoru Jeleni Kovačević, mentoru tokom izrade projekta Robertu Pečkaji Kovaču kao i svojim saradnicima Goranu Babiću i Miroslavu Malku što su uvek bili dostupni i spremni da pomognu.

Takođe se zahvaljujem svojoj porodici i priateljima na konstantnoj podršci tokom studiranja i izrade ovog rada.



УНИВЕРЗИТЕТ У НОВОМ САДУ

ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



SADRŽAJ

1. Uvod	1
2. Teorijske osnove.....	2
2.1 Višekanalni surround audio sistem.....	2
2.2 Sound bar sistemi.....	4
2.3 Virtualizacija surround zvučnika HRTF metodom	5
3. Koncept rešenja	7
3.1 Tok izrade projekta.....	7
3.1.1 Model 1	8
3.1.2 Model 2	8
3.1.3 Model 3	8
3.1.4 Završni model	8
3.2 Opis algoritma	9
3.3 Karakteristike digitalnog signal procesora	9
3.4 Razvojno okruženje	11
4. Implementacija	14
4.1 Analiza referentnog modela.....	14
4.2 Implementacija u asemblerском jeziku (Model 3)	15
4.2.1 Glavna funkcija modula (X_dap_ht_handle_speaker_virtualization)	15



УНИВЕРЗИТЕТ У НОВОМ САДУ

ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



4.2.2	Funkcije za podešavanje kanala na 5.1 ili 5.1.2 format	15
4.2.2.1	I_S_dap_ht_pvt_make_5_1_2	16
4.2.2.2	I_S_dap_ht_pvt_make_5_1	16
4.2.3	Funkcija za generisanje kompleksnih koeficijenata (I_S_sv_decorr_coefs_gen).....	17
4.2.4	Funkcija za obradu kanala koeficijentima (I_S_sv_decorr_process)	17
4.2.5	Provera ispravnosti funkcija u asemblerskom jeziku.....	17
4.3	Integracija modula u projekat (završni model).....	17
5.	Rezultati.....	19
5.1	Potrošnja resursa u modulu za virtualizaciju.....	19
5.2	Ručni bit-identični testovi	20
6.	Zaključak	22
7.	Literatura	24

SPISAK SLIKA

Slika 2.1 Audio sistem sa 5.1 konfiguracijom zvučnika	2
Slika 2.2 Audio sistem sa 5.1.2 konfiguracijom zvučnika	3
Slika 2.3 Audio sistem sa 7.1 konfiguracijom zvučnika	3
Slika 2.4 Refleksija zvuka gornjih zvučnika u sound bar sistemu	4
Slika 2.5 Reprodukcija 5.1.2 surround zvuka pomoću sound bar-a.....	5
Slika 2.6 OSD koncept	6
Slika 3.1 Tok izrade DSP aplikacije.....	7
Slika 3.2 Blok dijagram algoritma	9
Slika 3.3 Blok dijagram procesora CS49844	10
Slika 3.4 Tokovi podataka i akumulatorska jedinica DSP jezgra	11
Slika 3.5 Kontrolisano izvšavanje u CLIDE razvojnog okruženju	12
Slika 3.6 Sprega modula sa operativnim sistemom.....	13
Slika 5.1 Ručno poređenje izlaza referentnog i simulatorskog projekta.....	21

SPISAK TABELA

Tabela 3.1 Raspoloživa memorija procesora CS49844	10
Tabela 4.1 Pregled hijerarhije funkcija u modulu	15
Tabela 4.2 Kanali prisutni u 7.1.2 sistemu	16
Tabela 5.1 Utrošak MIPS-a za različite testne tokove	20
Tabela 5.2 Potrošnja memorije (broj reči)	20

SKRAĆENICE

DSP	- <i>Digital Signal Processing</i> , Digitalna obrada signala
LFE	- <i>Low- Frequency Effects</i> , Efekti niskih frekvencija
HD	- <i>High-definition</i> , Visoka definicija
HRTF	- <i>Head-related transfer function</i> , Prenosna funkcija koja se odnosi na glavu
ICC	- <i>Interaural crosstalk cancellation</i> , Poništenje zvuka koji dolazi od jednog do drugog uha
OSD	- <i>Optimal Source Distribution</i> , Optimalna distribucija izvora
CLIDE	- <i>Cirrus Logic Integrated Development Environment</i> , Cirrus Logic integrisano razvojno okruženje
MAC	- <i>Multiply And Accumulate</i> , Pomnoži i dodaj
SRS	- <i>Shifter/Rounder/Saturator</i> , Pomerač/Zaokruživač/Zasićivač
MIF	- <i>Module Interface</i> , Sprežni podsistem modula
MCV	- <i>Module Control Vector</i> , Tabela konfiguracionih parametara
MCT	- <i>Module Call Table</i> , Tabela rutina za spregu sa OS-om
OS	- <i>Operating System</i> , Operativni sistem
HQMF	- <i>Hybrid complex quadrature mirror filter bank</i> , Hibridna kompleksna kvadratna reflektujuća filter banka

1. Uvod

Zadatak ovog rada je implementacija bloka obrade za virtualizaciju surround zvučnika na DSP procesoru firme Cirrus Logic za sound bar sisteme.

Rad obuhvata upoznavanje sa osnovama audio DSP obrade i upoznavanje sa ciljnom platformom. Implementacija obuhvata prilagođenje referentnog C koda aritmetici ciljne DSP platforme i realizaciju u asemblerskom jeziku.

Cilj rada je upoznavanje sa pisanjem programske podrške u realnom vremenu za audio DSP aplikacije na ciljnoj platformi. Takođe, cilj je i upoznavanje sa opštim postupcima ispitivanja i verifikacije audio DSP aplikacija i izradom tehničke dokumentacije. Zadatak se oslanja na rad u programskim jezicima C i Cirrus Logic asembler.

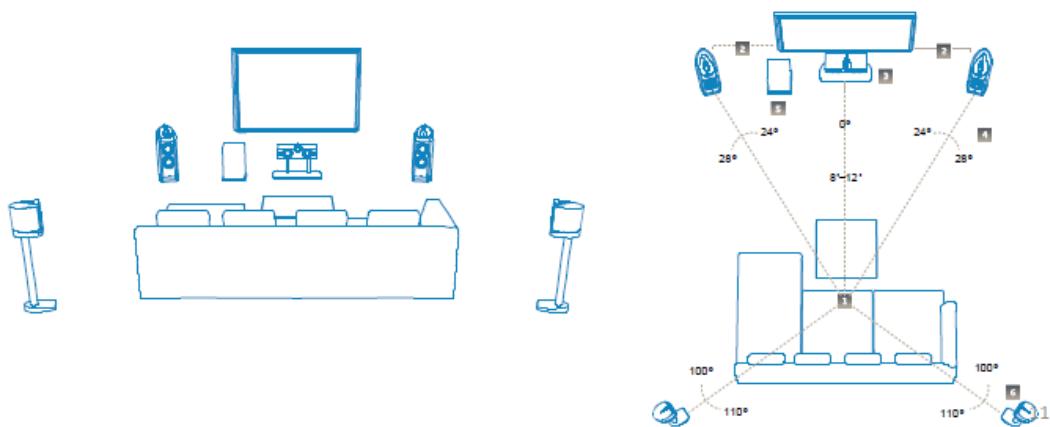
Rad se sastoji iz sedam poglavlja. U drugom poglavlju je opisan pojam višekanalnih surround audio sistema, sound bar sistema, kao i metoda virtualizacije surround zvučnika koja je korišćena u projektu. Treće poglavlje predstavlja koncept rada kroz faze izrade projekta. Takođe je dat i kratak opis algoritma, uz osvrt na ciljnu platformu i razvojno okruženje u kome je izrađen asemblerski projekat. Četvrto poglavlje opisuje implementaciju algoritma nabranjem glavnih funkcija. U petom poglavlju su date vrednosti utroška memorije i procesorskog vremena, kao i rezultati ručnih bit-identičnih testova. Poslednja dva poglavlja predstavljaju zaključak i literaturu korišćenu tokom izrade projekta.

2. Teorijske osnove

2.1 Višekanalni surround audio sistem

Višekanalni audio sistemi su široko rasprostranjeni u modernim audio uređajima. Pojam višekanalni audio sistemi označava da su ovi sistemi sposobni da rukuju sa više audio kanala kako bi rekonstruisali zvuk preko nekoliko zvučnika.

Različite vrste sistema zvučnika su najčešće označene pomoću decimalne tačke (.) (2.1, 4.1, 5.1, 6.1, 7.1, itd.), gde korišćene cifre zavise od broja audio kanala. Neki audio sistemi imaju samo jedan kanal (mono) ili dva kanala (stereo zvuk). Na slici 2.1 je prikazan primer 5.1 audio sistema.



Slika 2.1 Audio sistem sa 5.1 konfiguracijom zvučnika

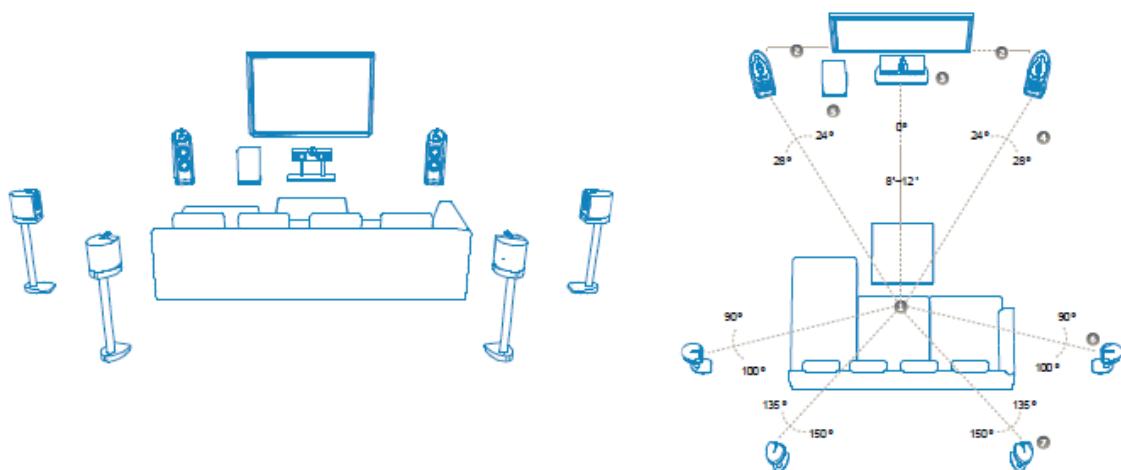
Prva cifra predstavlja broj primarnih kanala, od kojih se svaki reproducuje na jednom zvučniku punog opsega reprodukcije (koji mogu da rukuju frekvencijama od 100 Hz do 22

kHz), dok druga cifra predstavlja prisustvo LFE kanala. U novim sistemima se javljaju i oznake sa trećom cifrom, gde treća cifra označava prisustvo gornjih kanala (Slika 2.2)



Slika 2.2 Audio sistem sa 5.1.2 konfiguracijom zvučnika

Surround zvuk je pojam koji opisuje audio izlaz koji se čini da okružuje (eng. surround) slušaoca za 360 stepeni, tj. dobija se utisak da zvuk dolazi iz svih pravaca. To se realizuje tako što zvučnici raspoređeni na različitim mestima u prostoriji primaju višekanalni audio. Na slici 2.3 je prikazan audio sistem nove generacije sa sedam primarnih kanala i jednim LFE kanalom [4].



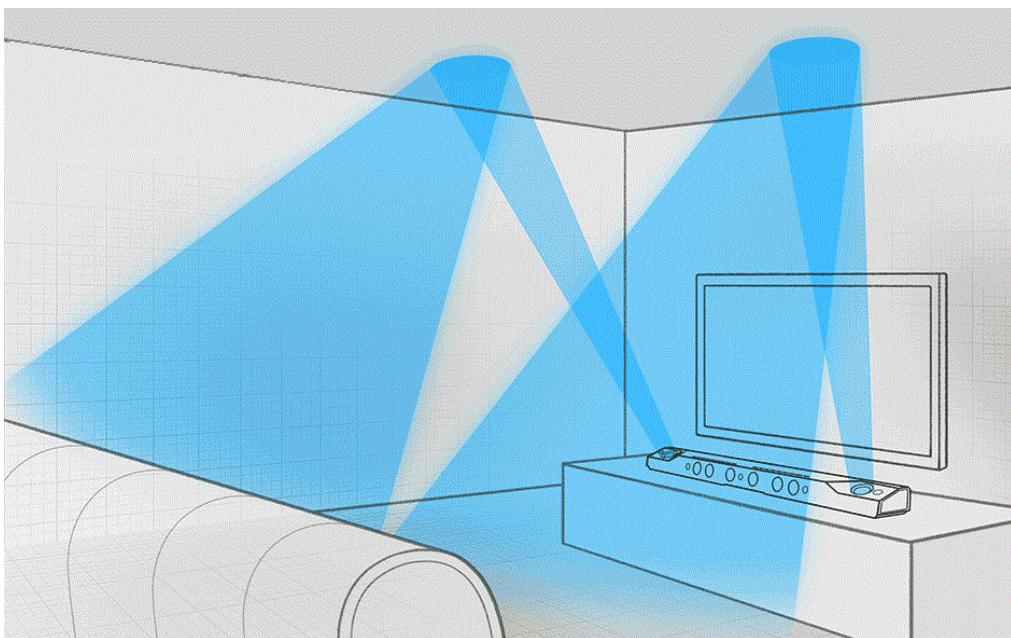
Slika 2.3 Audio sistem sa 7.1 konfiguracijom zvučnika

2.2 Sound bar sistemi

Sa pojavom HD (High definition, visoka definicija) televizora sa ravnim ekranom, ugrađeni zvučnici su postajali sve slabiji i sound bar sistemi su predstavljeni kao najbolji način za unapređenje zvuka, kako bi zvuk dostigao visok kvalitet slike. Međutim, doseg mogućnosti sound bar sistema se tu završavao.

Dodavanjem naprednih dekodera i mogućnosti virtualizacije zvuka, kako bi se stvorio sveobuhvatniji utisak nego što su pružali klasični 2.0 ili 2.1 sound bar sistemi, sound bar sistemi su vremenom napredovali. Oni su prepoznati kao najbolja ponuda korisnicima koji nemaju mogućnost da upgrade surround sistem zvučnika, i danas odgovarajućim metodama virtualizacije pružaju surround efekat koji je skoro dostigao nivo fizičkog surround sistema.

Novim znanjima iz psihoakustike i zvučne fizike je razvijena tehnika koja može da pusti zvuk i iznad slušaoca, pomoću zvučnika koji se nalaze samo nekoliko stopa od poda. Ova znanja su dalje razvijena za potrebe integracije gornjih zvučnika u sound bar sisteme, čiji zvuk se odbija o plafon i tako daje korisniku utisak da zvuk dolazi iz izvora iznad njega (Slika 2.4)



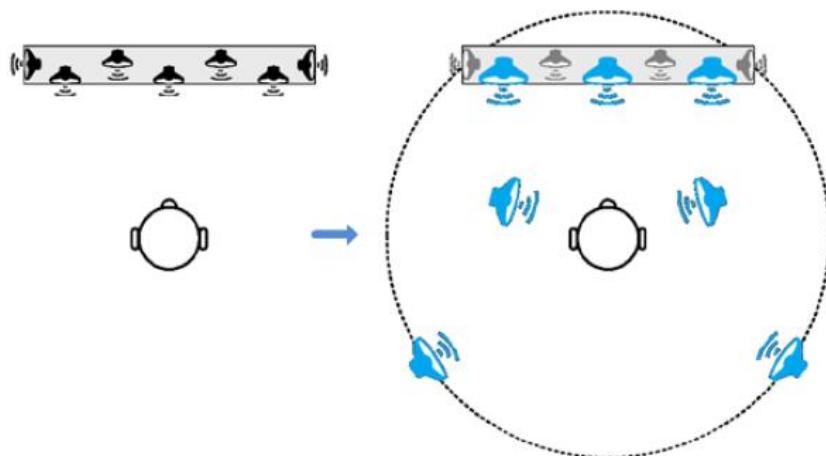
Slika 2.4 Refleksija zvuka gornjih zvučnika u sound bar sistemu

U nastavku poglavlja će biti opisane metode virtualizacije, koje uz refleksiju koju omogućava pozicija zvučnika, pružaju surround efekat puštanjem zvuka preko zvučnog sistema koji se nalazi ispred slušaoca.

2.3 Virtualizacija surround zvučnika HRTF metodom

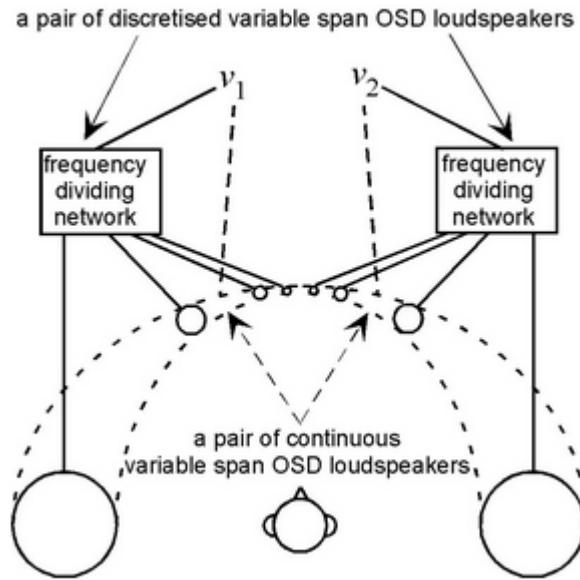
Klasičan surround audio sistem ima nekoliko ograničenja. Prvenstveno, bez obzira na pokušaje da se postigne disperzija zvuka, i dalje je slušaocu očigledno da zvuk dolazi iz zvučnika, a ne iz okruženja. Takođe, zvuk ne okružuje samog slušaoca, već prostor oko koga su raspoređeni zvučnici. Na kraju, čovek ne može da pravilno locira zvuk koji se kreće između prednjih i zadnjih zvučnika, jer se zvuku u tom prostoru menja samo amplituda.

U virtualnom sistemu (slika 2.5) ne postoje ova ograničenja. Slušni i moždani sistemi su stimulisani pomoću prenosnih funkcija baziranih na HRTF (Head-related transfer function, prenosna funkcija koja je bazirana na glavi slušaoca), i realističnost iskustva može biti značajno povećana. Međutim, i u ovom slučaju postoje dva ograničenja: kvalitet zvuka i prava pozicija slušaoca [3].



Slika 2.5 Reprodukcija 5.1.2 surround zvuka pomoću sound bar-a

Primenom HTRF obrade, virtualizacioni sistemi koriste ICC (Interaural crosstalk cancellation, poništenje zvuka koje dolazi od jednog do drugog uha) tehniku. Ova tehnika izoluje zvuk sa levog zvučnika do levog uha, i sa desnog zvučnika do desnog uha, što iako je efikasno, stvara veoma uzak prostor u kome se dobija željeni surround efekat. Dodavanjem manipulacije vremenom i fazama, taj se prostor dodatno sužava. Iz tog razloga se koriste dodatne tehnike, kako bi se izbegle pomenute metode i tako proširio prostor unutar kog slušalac ima utisak surround zvuka. Najčešće korišćen koncept za postizanje tog efekta je OSD (Optimal Source Distribution) (Slika 2.6)



Slika 2.6 OSD koncept

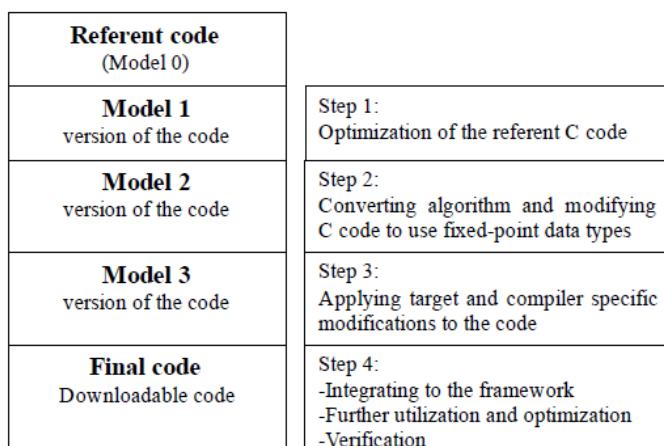
Inverzni filteri imaju ujednačen frekvencijski odziv tako da se javlja mala neželjena promena usled različitih HRTF metoda kao i različitih pozicija u prostoriji. Pojačanje zvuka je reprodukovano svuda u prostoriji, čak i kod stereo i surround snimaka. Za razliku od ICC tehnike, efekat virtualizacije je nezavisan od frekvencije, i to omogućava prezentaciju signala do levog i desnog uha mnogobrojnih slušalaca u konstantnim intervalima, u zavisnosti od pozicije u prostoriji. Sistem je uz to otporan i na refleksiju u prostorima koji odjekuju [5].

Pod podobnim uslovima, višekanalni virtualizatori su superiorniji u odnosu na surround sistem zvučnika po dvema osnovama. Zahtevaju samo dva zvučnika ili sound bar, kako bi stvorili iskustvo bolje nego što to uspeva višezvučni surround sistem. Takođe, oni dostavljaju zvuk do velikog broja potrošača koji nisu u mogućnosti ili ne žele da nabave surround sistem. Daljim napredovanjem virtualizacije, kroz stimulaciju ljudskog slušnog i moždanog sistema, pre nego kroz standardne uređaje, dobiće se realistično iskustvo sa očuvanim kvalitetom zvuka.

3. Koncept rešenja

3.1 Tok izrade projekta

Metodologija izrade aplikacija za DSP platformu predstavlja iterativan postupak, koji se sastoji iz nekoliko faza (Slika 3.1). Tokom faza se početna verzija koda razvija u semantički ekvivalentne verzije, koje su svakim korakom sve bliže završnom rešenju. Referentni model se dobija od onog ko razvija aplikaciju i rezultati izvršenja ovog modela smatraju se konačnim i ispravnim. Tačnost rezultata svakog od sledećih modela se proverava poređenjem sa prethodnim. Kad se napravi bilo kakva izmena nad referentnim modelom, on postaje Model 1. Model 1 je formiran uvođenjem funkcionalnih optimizacija. Model 2 podrazumeva prilagođavanje koda aritmetici ciljne arhitekture. Model 3 predstavlja potpuno prevodiv C kod za namensku platformu. Korak nakon uspešno formiranog Modela 3 jeste detaljna procena utroška resursa. Nakon toga, ukoliko su rezultati prethodne faze zadovoljili kriterijume, sledi “ugradnja” tog koda u postojeće programsko okruženje. [7]



Slika 3.1 Tok izrade DSP aplikacije

3.1.1 Model 1

Početna faza izrade aplikacije je funkcionalna optimizacija referentnog koda. Tokom pisanja referentnog koda se najviše pažnje posvećuje ispravnoj implementaciji algoritma. Ne vodi se računa o optimizovanosti koda, kao ni o specifikacijama ciljne platforme. Iz tog razloga je moguće izvršiti određeni broj promena u prvom modelu, među kojima su zamena indeksnog adresiranja nizova adresiranjem preko pokazivača, smanjenje broja argumenata u funkciji i prilagođavanje petlji kasnije korišćenim hardverskim petljama. Nakon ovih izmena je moguće preći na izradu drugog modela.

3.1.2 Model 2

Referentni kod DSP aplikacije može da sadrži obradu podataka koristeći aritmetiku sa pokretnim ili nepokretnim zarezom, u zavisnosti od toga za koju fizičku arhitekturu je pisan. S obzirom da sam C standard ne sadrži podršku za tipove u nepokretnom zarezu, način na koji je ta aritmetika podržana zavisi od kompjlera za koji je C kod napisan. Postoji proširenje C standarda za namenske procesore (Embedded C standard), koje između ostalog sadrži tipove podataka u nepokretnom zarezu i operacije nad njima. Kod kompjlera koji podržavaju ovo proširenje C standarda ti tipovi se mogu koristiti direktno, dok se kod ostalih kompjlera obrada koja koristi aritmetiku u nepokretnom zarezu najčešće vrši upotreboom celobrojnih tipova. Cilj prilikom izrade Modela 2 jeste prilagođenje koda ograničenjima i mogućnostima ciljne platforme po pitanju aritmetike.

3.1.3 Model 3

Model 2 predstavlja i prvu verziju trećeg modela koja se prebacuje u samostalni CLIDE projekat, kako bi algoritam mogao da se izvrši na Coyote simulatoru ili na Coyote razvojnoj ploči. Nakon ovog koraka je moguće proveriti ispravnost modela koristeći asemblerski debager, koji povezuje C instrukcije sa njima odgovarajućim asemblerskim instrukcijama. Model 3 je prva verzija koja se može u potpunosti prevesti u asemblersku datoteku. U okviru izrade ovog modela je moguće sprovesti i dalje optimizacije u vidu korišćenja inline asemblera ili pisanja funkcija u potpunosti u asemblerskom jeziku. Nakon uspešno završene optimizacije je još potrebno ugraditi kod u programsko okruženje za odgovarajuću platformu.

3.1.4 Završni model

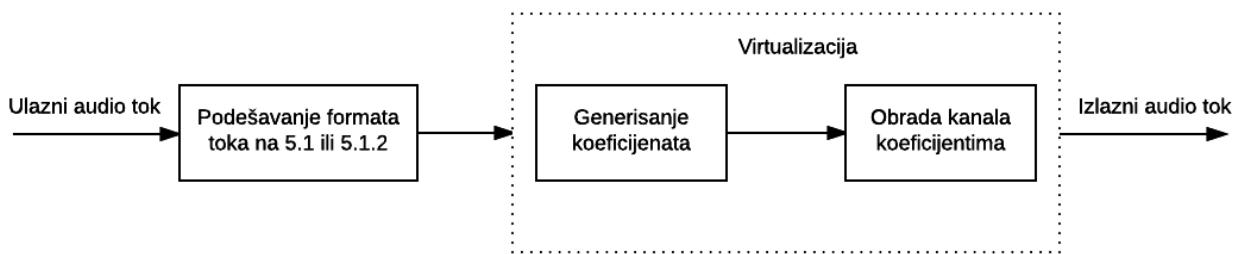
U okviru trećeg modela se dobija ispravan i funkcionalan algoritam, ali sam kod ne može biti pušten preko DSP platforme. Potrebno je još ugraditi nekoliko elemenata, koji zavise od platforme i koji rukuju neophodnim resursima i pružaju okruženje za puštanje na

platformi. Za jednu platformu je te elemente potrebno napraviti jednom, i moguće ih je kasnije ponovo koristiti. U završnom modelu se takođe mogu uraditi još neke optimizacije vezane za ograničenja resursa. Kada se završni kod uspešno pusti na DSP platformi, vrše se procesi verifikacije.

3.2 Opis algoritma

Algoritam za virtualizaciju surround zvučnika generiše 5.1 ili 5.1.2 virtualizovane audio tokove, obradom prosleđenih audio tokova u proizvoljnom formatu. Dodatno, algoritam pojačava kanale surround zvučnika, kako bi se stvorio realniji surround efekat.

Prvi korak u algoritmu predstavlja podešavanje ulaznih kanala na 5.1 ili 5.1.2 format, smanjivanjem ili povećavanjem broja kanala zavisno od ulaznog formata audio toka. Ovom obradom se dobija format bafera kompatibilan sa virtualizatorom. Nakon toga se vrši virtualizacija nad odgovarajućim parom kanala individualnih zvučnika. Proces virtualizacije počinje računanjem odgovarajućih kompleksnih koeficijenata. Računanje koeficijenata za virtualizaciju je bazirano na naprednoj HRTF i ICC metodi, opisanim u drugom poglavlju. Prenosna funkcija zasnovana na glavi slušaoca zahteva kao ulazne parametre uglove od slušaoca do fizičkog i virtualnog izvora zvuka. Generisanim koeficijentima se obrađuju levi i desni surround kanali, a za 5.1.2 izlazni format audio toka i levi i desni gornji surround kanali. Puštanjem virtualizovanih kanala preko zvučnika ispred slušaoca se simulira njihova prostorna pozicija (Slika 3.2)

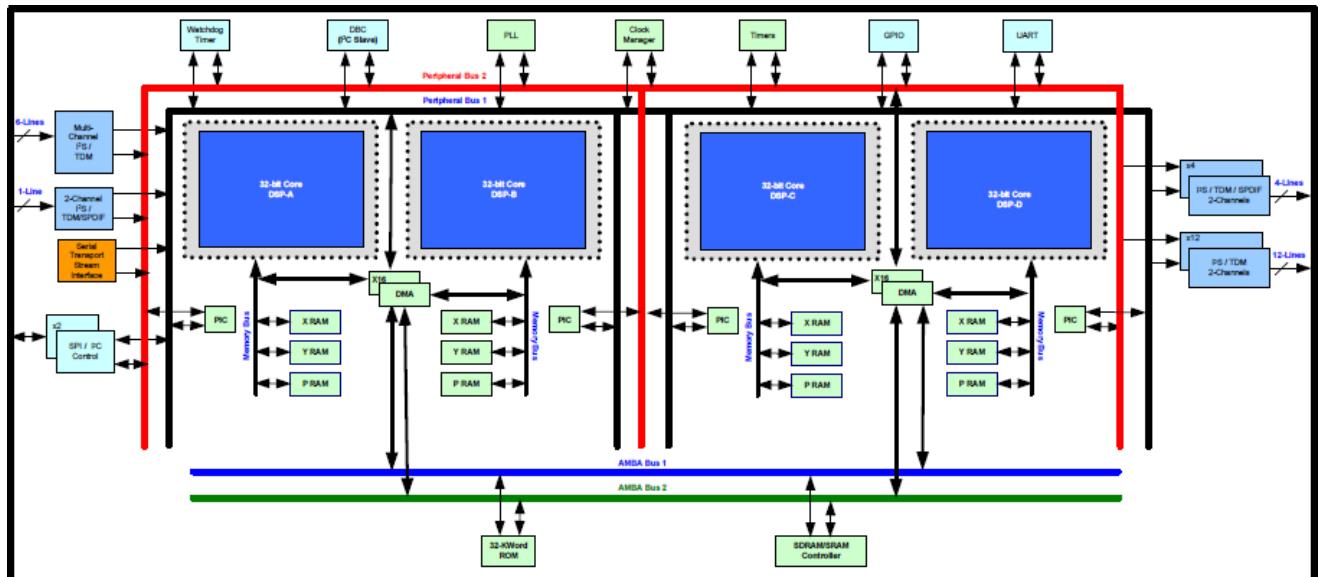


Slika 3.2 Blok dijagram algoritma

3.3 Karakteristike digitalnog signal procesora

Projekat u koji je integriran modul za virtualizaciju surround zvučnika se izvršava na platformi na kojoj se nalazi DSP procesor CS49844 kompanije Cirrus Logic (Slika 3.3), koji sadrži četiri 32-bitna DSP jezgra (Tabela 3.1). Ovaj digitalni signal procesor sa aritmetikom

nepokretnog zareza može da izvrši dve MAC (Multiply And Accumulate, Pomnoži i dodaj) operacije po taktu.



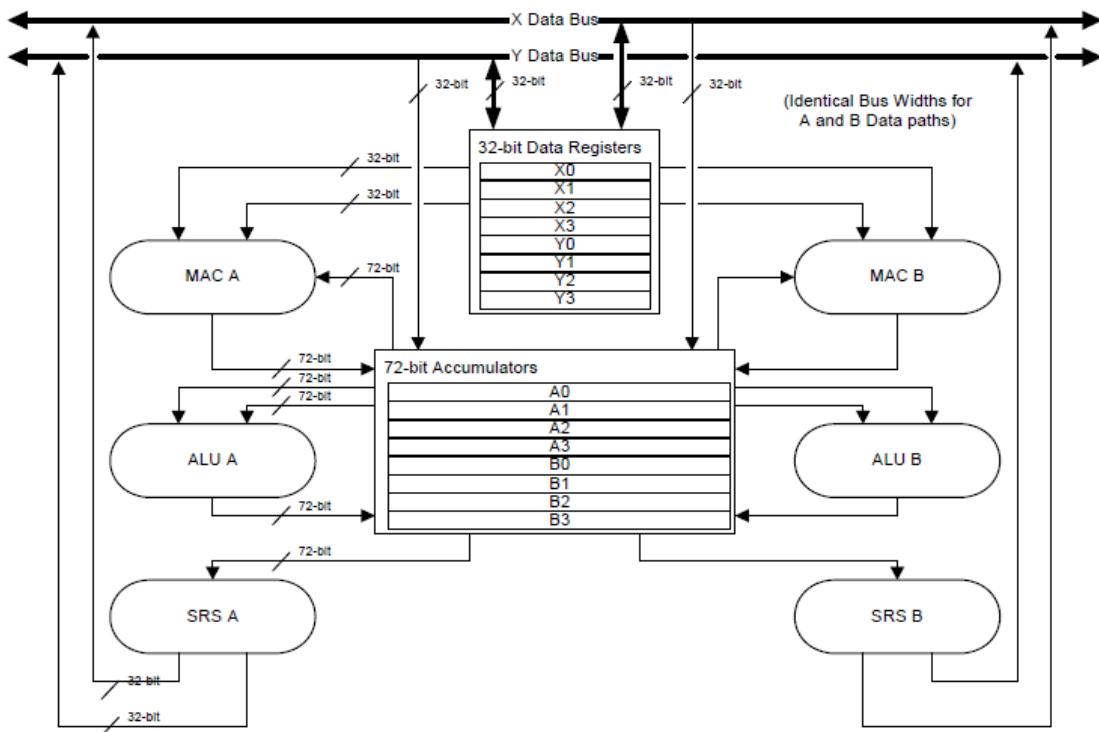
Slika 3.3 Blok dijagram procesora CS49844

Tip memorije	Jezgro A	Jezgro B	Jezgro C	Jezgro D
X,Y,P	60kWord SRAM	60kWord SRAM	60kWord SRAM	60kWord SRAM

Tabela 3.1 Raspoloživa memorija procesora CS49844

Četiri 32-bitna DSP jezgra se zasnivaju na unapređenoj Harvard arhitekturu, što znači da poseduju posebnu memoriju za podatke i posebnu memoriju za instrukcije [1]. Memorija za podatke je dodatno podeljena u dve posebne celine.

Svako jezgro se sastoji iz jedinice za kontrolu toka programa, dva paralelna toka podataka (A i B) i paralelnih jedinica za generisanje adresa (AGU). Jedinica za generisanje adresa sadrži dvanaest indeksnih registara i dvanaest odgovarajućih modulo registara koji omogućavaju različite adresne režime [6]. Slika 3.4 prikazuje protok podataka kroz tok podataka i akumulatorsku jedinicu.



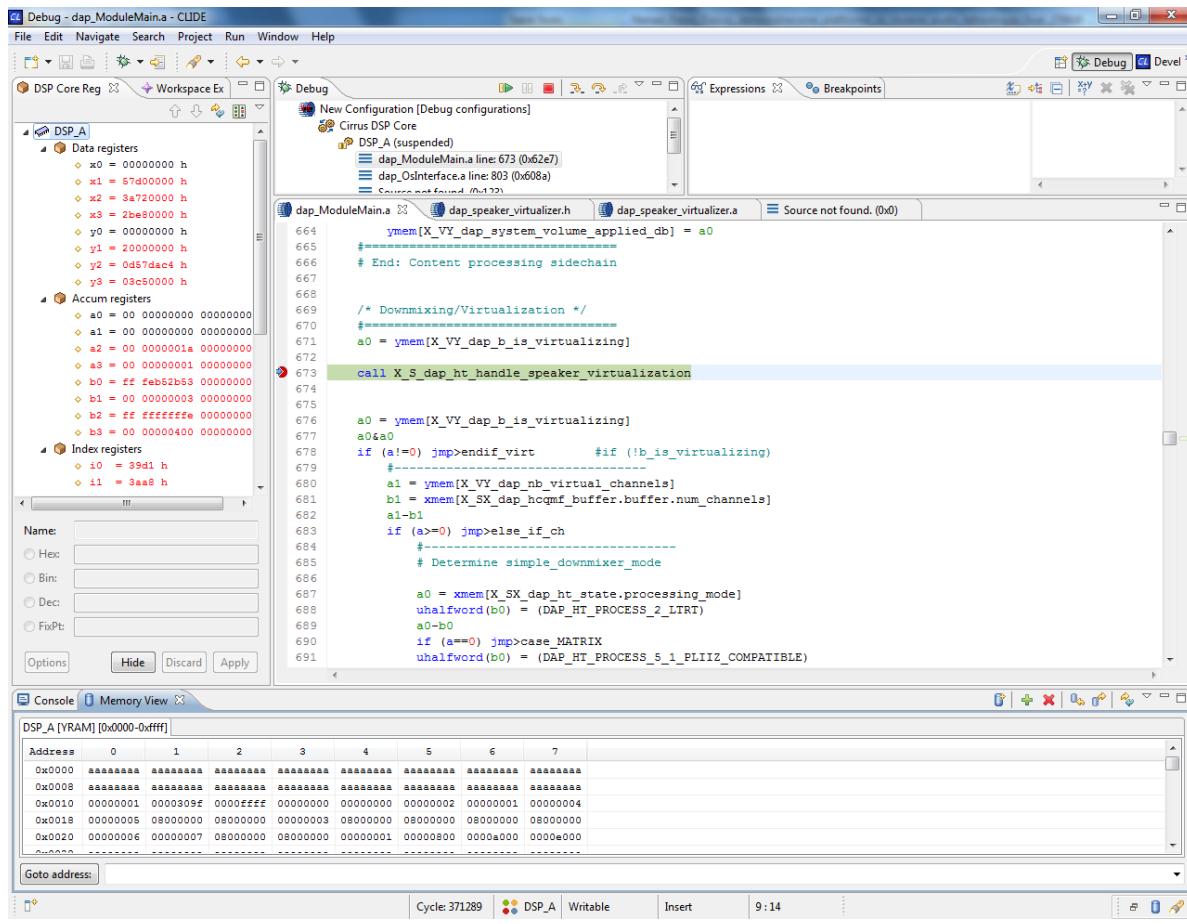
Slika 3.4 Tokovi podataka i akumulatorska jedinica DSP jezgra

Svaki tok podataka raspolaže sa četiri 32-bitna registra opšte namene i četiri 72-bitna akumulatora. Svaki akumulator se sastoji iz tri nadovezana registra označenih kao Guard (8 bita), High (32 bita) i Low (32 bita), gde se svakom od njih može posebno pristupiti. Takođe, oba toka podataka raspolažu i sa nezavisnom MAC, SRS (Shifter/Rounder/Saturator, Pomerač/Zaokruživač/Zasićivač) i aritmetičko-logičkom jedinicom (ALU). ALU jedinica je odgovorna za sve logičke operacije izvršene nad akumulatorima.

3.4 Razvojno okruženje

Za razvoj asemblerorskog koda korišćen je CLIDE, integrisano razvojno okruženje za razvoj programske podrške za DSP procesore proizvođača Cirrus Logic (Coyote familija procesora). CLIDE razvojno okruženje zasnovano je na Eclipse platformi. Eclipse je široko rasprostranjen kao osnova mnogih razvojnih okruženja, jer je otvorenog koda i nudi mnoštvo korisnih alata za razvoj programske podrške. CLIDE proširenja u Eclipse-u su prilagođena radu na DSP procesorima proizvedenim od strane Cirrus Logic-a.

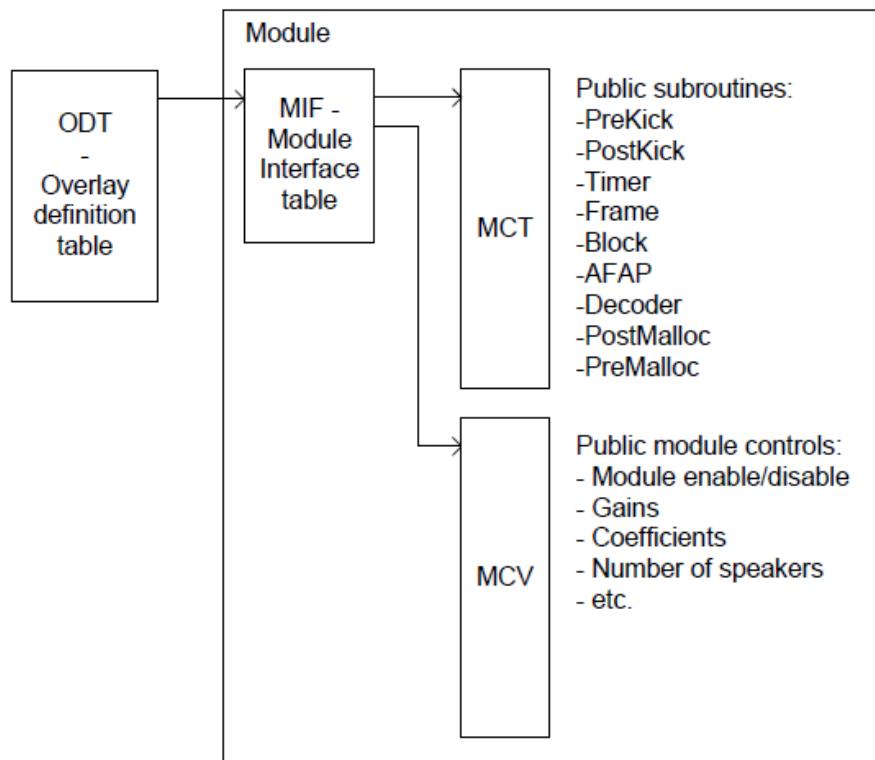
CLIDE razvojno okruženje (Slika 3.5) obezbeđuje potpuno funkcionalne uređivače teksta za C ili asemblerSKI kod, kao i mogućnost prevođenja aplikacije i podršku za smeštanje i spuštanje DSP aplikacije na razvojnu ploču. Rad značajno olakšava mogućnost korišćenja alata za kontrolisano izvršavanje programa, uz opcije kao što su pregled memorije i stanja registara.



Slika 3.5 Kontrolisano izvšavanje u CLIDE razvojnom okruženju

Cirrus Logic programsko okruženje – framework predstavlja sistemsku programsku podršku procesora koja skraćuje vreme i uloženi rad za razvoj aplikacije uvodeći neke od ideja i metodologija iz objektno orijentisanog programiranja u svet asemblerorskog koda. Jezgro programskog okruženja se sastoji od jednostavnog operativnog sistema (OS) čija je glavna uloga da bude raspoređivač za određen broj procesnih entiteta (modula). OS predstavlja monitorsku petlju koja poziva rutine odgovarajućih modula po unapred definisanom redosledu [2].

Svaki modul ima svoj jedinstveni sprežni podsistem (Module Interface – MIF) kojim je modul povezan sa OS-om. Njega čini MIF tabela koja sadrži pokazivače na tabele sa ostalim sprežnim informacijama. Dve najvažnije tabele su MCT tabela (Module Call Table), i MCV tabela (Module Control Vector). Sa OS strane sprega ka modulima se sastoji od ODT tabele (Overlay Definition Table) koja sadrži pokazivače na MIF tabele svih učitanih modula (Slika 3.6)



Slika 3.6 Sprega modula sa operativnim sistemom

MCT tabela je sastavljena od pokazivača na javne rutine. Redosled elemenata u tabeli je unapred definisan, a ukoliko neka od rutina nije definisana, na mestu njenog pokazivača se nalazi nula. Ove rutine OS poziva kao odgovor na pojavu odgovarajućih događaja u sistemu, i to su jedine rutine modula kojima OS može pristupiti. Svaka od njih ima predefinisanu namenu i njihovim pozivanjem od strane OS-a, modul odgovara na određeni događaj u sistemu. MCV tabela sadrži niz javno dostupnih konfiguracionih parametara datog modula. MCV omogućava konfigurisanje modula od strane glavnog kontrolera uređaja. Ova tabela nema unapred definisanu strukturu, već programer formuliše njen sadržaj i strukturu.

4. Implementacija

Zadatak projekta je implementacija bloka obrade za virtualizaciju surround zvučnika na DSP procesoru firme Cirrus Logic za sound bar sisteme. Pri izradi modula je korišćena metodologija opisana u prethodnom poglavlju, uz nekoliko izmena. Pošto referentni kod u okviru ovog projekta ne sadrži samo algoritam, već je optimizovan i podržava aritmetiku nepokretnog zareza, referentni model je po metodologiji na nivou Modela 2. Iz tog razloga je prvi korak analiza referentnog koda, umesto sastavljanja Modela 1. Takođe, kako nije potrebno od završenog trećeg modela napraviti samostalan projekat u C programskom jeziku, već je samostalan projekat pisan direktno u asemblerском jeziku, izrada Modela 3 predstavlja implementaciju direktno u asemblerском jeziku. Izrada završnog modela nije izmenjena u odnosu na izloženu metodologiju. Pošto integriran projekat nije bio puštan na platformi, merenje i verifikacija su vršeni u okviru samostalnog projekta. Precizniji tok izrade projekta će biti opisan u nastavku poglavlja, kroz faze izrade modela.

4.1 Analiza referentnog modela

Prva faza u izradi projekta je analiza referentnog modela. Algoritam za virtualizaciju surround zvučnika napisan u programskom jeziku C u okviru Visual Studio projekta predstavlja ispravan i optimizovan referentni kod u aritmetici nepokretnog zareza, na osnovu koga je urađen modul u asembleru. Pre izrade svake pojedinačne funkcije, analizirana je funkcija u okviru referentnog koda sa akcentom na razumevanje algoritma. Takođe, analizirane su povratne vrednosti funkcije ili izmene koje one unosi, kao i korišćene strukture podataka.

4.2 Implementacija u asemblerskom jeziku (Model 3)

Nakon analize referentnog koda, počinje se sa realizacijom samostalnog projekta u asemblerskom jeziku. Već tokom početnog pisanja funkcija se vodi računa o iskorišćenju resursa i mogućnosti paralelizacije. Na mestima gde je to bilo moguće nisu stvarane memorijske strukture, već su korišćeni registri podataka ili akumulatori. Najveći izazov u ovoj fazi je ispravno prebacivanje algoritma nepokretnog zareza u C programskom jeziku u algoritam u asembleru, pre svega zbog aritmetičkih operacija čiji rezultat je neophodno normalizovati ili šiftovati. Nakon završetka i provere prve verzije funkcije, rađene su dalje optimizacije dodavanjem većeg broja paralelnih operacija. U nekim slučajevima su rađene male izmene algoritma, ukoliko je to dovodilo do velike uštete ciklusa. Nakon završetka ove faze se dobija optimizovan i ispravan modul u asemblerskom jeziku.

4.2.1 Glavna funkcija modula (X_dap_ht_handle_speaker_virtualization)

Datom funkcijom se poziva modul za virtualizaciju surround zvučnika. To je jedina funkcija koja se vidi iz glavnog projekta i ostalih modula. U okviru nje se dalje pozivaju unutrašnje funkcije koje u sebi sadrže implementaciju algoritma za virtualizaciju (Tabela 4.1). U nastavku su opisane najbitnije unutrašnje funkcije za programsко rešenje.

```
X_dap_ht_handle_speaker_virtualization
• I_S_dap_ht_pvt_make_5_1
• I_S_dap_ht_pvt_make_5_1_2
• I_S_dap_speaker_virtualizer_process_channel
  ○ I_S_sv_decorr_coefs_gen
    ■ I_S_sv_virtcoefs_symmetric_init
      ♦ I_S_hsf_get_config
      ♦ I_S_vsf
  ○ I_S_sv_decorr_process
```

Tabela 4.1 Pregled hijerarhije funkcija u modulu

4.2.2 Funkcije za podešavanje kanala na 5.1 ili 5.1.2 format

U obe funkcije je kao ulazni parametar prosleđena HCQMF (Hybrid complex quadrature mirror filter bank, Hibridna kompleksna kvadratna reflektujuća filter banka) struktura, koja sadrži konfiguraciju bafera sa audio kanalima (na primer broj kanala, prisustvo gornjih surround kanala) kao i sam bafer. U okviru tog bafera se nalaze kanali raspoređeni u unapred definisanom redosledu (Tabela 4.2)

Naziv kanala
L (Left)
R (Right)
C (Center)
LFE (Low Frequency Effect)
Ls (Left Surround)
Rs (Right Surround)
Lb (Left Back)
Rb (Right Back)
Lts (Left Top Surround)
Rts (Right Top Surround)

Tabela 4.2 Kanali prisutni u 7.1.2 sistemu

4.2.2.1 I_S_dap_ht_pvt_make_5_1_2

Ova funkcija na izlazu daje 5.1.2 bafer, koji je kompatibilan sa virtualizatorom. Ako je ulaz u već traženom formatu, ne radi se nikakva modifikacija. U slučaju kad je prosleđen format ulaza 5.1, dva gornja surround kanala se ispunjavaju nulama. Ako je ulazni format 7.1 broj surround kanala se smanjuje na pet, i kao u prethodnom slučaju se gornji surround kanali ispunjavaju nulama. Na kraju, ako je format ulaza 7.1.2, 7.1 format se smanjuje na format 5.1, dok se gornji surround kanali prosleđuju kakvi jesu. Algoritam za smanjivanje broja surround kanala je preuzet iz glavnog projekta.

4.2.2.2 I_S_dap_ht_pvt_make_5_1

Slično kao i prethodna funkcija, podešava se format izlaznog bafera, ali u ovoj funkciji na 5.1 format. U slučaju kad je prosleđen 5.1 format, ne radi se nikakva modifikacija. Kad je ulaz u formatu 7.1, broj surround kanala se smanjuje na pet. Ulaznom baferu formata 7.1.2 se takođe broj surround kanala smanjuje na pet, i prosleđuje se izostavljanjem dva gornja surround kanala. Na kraju, ulaz formata 5.1.2 se prosleđuje uz izostavljanje dva gornja surround kanala.

4.2.3 Funkcija za generisanje kompleksnih koeficijenata (I_S_sv_decorr_coefs_gen)

Data funkcija, uz pomoć nekoliko unutrašnjih funkcija generiše set kompleksnih koeficijenata koji se koriste za virtualizaciju. Neophodan ulazni parametar za ovu funkciju je ugao od slušaoca (relativan u odnosu na osu koja gleda napred) do mesta gde treba da bude lociran virtualan zvučnik. Takođe, prosleđuje se pola vrednosti ugla u stepenima, koji se formira od linija koje povezuju slušaoca i mesta gde se fizički nalaze zvučnici. Još jedan potreban podatak je niz koji sadrži centralizovane frekvencije svakog od virtualnih podopsega normalizovanih na brzinu takta. Na kraju funkcije se dobijaju nizovi koeficijenata koji se koriste za obradu ulaznog levog i desnog kanala kako bi se dobili izlazni levi i desni kanal, kao i nizovi koeficijenata kojima se od ulaznog levog i desnog kanala dobiju izlazni desni i levi kanal.

4.2.4 Funkcija za obradu kanala koeficijentima (I_S_sv_decorr_process)

U okviru ove funkcije se vrši virtualizacija kanala, njihovom obradom generisanim koeficijentima. Obrada se vrši nad prosleđenim levim i desnim surround kanalima, a u slučaju 5.1.2 formata bafera sa kanalima, i nad levim i desnim gornjim surround kanalima. Osim toga se prosleđuju i nizovi koeficijenata, čija je vrednost normalizovana. Virtualizacijom zvučnika može da se unese maksimalno pojačanje od 5.7dB. Završetkom ove obrade je prosleđen par kanala, koji puštanjem preko prednjih zvučnika daje zvuk koji simulira njihovu prostornu poziciju.

4.2.5 Provera ispravnosti funkcija u asemblerskom jeziku

Tokom rada na implementaciji modula u asemblerskom jeziku je rađena stalna provera ispravnosti algoritma, poređenjem asemblerske funkcije sa funkcijom referentnog koda. To je omogućeno pripremom tekstualne datoteka sa ulaznim parametrima funkcije u okviru referentnog projekta i njeno korišćenje u asemblerskom samostalnom projektu. Iako su početne provere radene jednostavnim poređenjem lokalnih promenljivih referentne funkcije i stanjem registara u asemblerskom projektu, da bi se testirale celokupne funkcije sa velikim brojem poziva, napravljene su izlazne binarne datoteke sa povratnim vrednostima funkcija ili podacima koje je funkcija izmenila. Te izlazne datoteke u referentnom i asemblerskom projektu su poređene uz očekivani rezultat od potpune jednakosti na nivou bita.

4.3 Integracija modula u projekat (završni model)

Nakon što je urađen modul u okviru samostalnog asemblerskog projekta, sa proverenim i optimizovanim funkcijama, bilo ga je potrebno integrisati u projekat sa ostalim modulima koji vrše razne obrade nad ulaznim audio kanalima. Posle unošenja svih funkcija i

struktura podataka, provereno je da li je količina korišćene memorije u granicama koje platforma dozvoljava. Još tokom izrade samostalnog projekta se moralo voditi računa o završnoj fazi implementacije, jer su strukture podataka koje se globalno koriste u okviru projekta morale na ispravan način da budu uključene i obrađene u modulu za virtualizaciju surround zvučnika. Ovde se koncept prosleđivanja struktura, kao što su baferi, u vidu pokazivača na njih pokazao kao najbolji način uniformisanja modula i mogućnosti, da i uz spoljašnje izmene, sam modul ostane ispravan.

5. Rezultati

5.1 Potrošnja resursa u modulu za virtualizaciju

Nakon završetka modula za virtualizaciju surround zvučnika je urađena procena utroška memorije DSP procesora kao i procesorskog vremena potrebnog za obradu u modulu za virtualizaciju. Utrošak memorije je izražen kroz broj zauzetih reči, odnosno 32-bitnih memorijskih lokacija. S druge strane, iako je u radnom okruženju moguće dobiti samo broj utrošenih ciklusa, za svaki od testnih slučajeva je pomoću sledeće formule izmeren utrošak procesorskog vremena u milionima instrukcija po sekundi:

$$MIPS = \frac{\text{broj_ciklusa} * \frac{Fs}{\text{BLOCK_SIZE}}}{1000000}$$

Parametar F_s predstavlja frekvenciju odabiranja, broj odbiraka u jednoj sekundi ulaznog signala, koja u projektu iznosi 48kHz. $BLOCK_SIZE$ predstavlja veličinu bloka obrade, koja iznosi 256 odbiraka. Kao što je već napomenuto, tokom kontrolisanog izvršenja programa je moguće utvrditi tačan broj ciklusa potreban za izvršavanje ovog modula. U prvoj koloni Tabele 5.1 je prikazan utrošak procesorskog vremena. Prosečan utrošak je računat kao srednja vrednost utroška procesora dobijena kontrolisanim izvršavanjem modula pedeset puta na simulatoru. Vreme izvršavanja testova prikazano u drugoj koloni tabele obuhvata trajanje izvršavanja pedeset prolaza funkcije za virtualizaciju, uključujući i vreme potrebno za učitavanje ulaznih podataka iz binarnih datoteka i za upisivanje krajnjih vrednosti u binarnu datoteku. Zbog nemogućnosti korišćenja preciznih metoda merenja vremena, rezultati su podložni greškama.

Testni tokovi	Prosečan utrošak procesorskog vremena [mips]	Vreme izvršavanja testova [s]
sys_logic_ngcs_bypass_spk_i7d1d2_o5d1d2_io48k	8,63	16,9
sys_logic_ngcs_outputmode_spk_i7ch1_o5d1d2_io44k1	7,79	16,3
sys_logic_ngcs_bypass_spk_io5d1d2_io44k1	7,79	16,3
sys_logic_ngcs_outputmode_spk_i5ch1_o5d1d2_io48k	7,89	16,4
sys_logic_ngcs_bypass_spk_i7d1d2_o5ch1_io48k	5,17	12,5
sys_logic_ngcs_outputmode_spk_i7ch1_o5ch1_io44k1	4,39	11,9
sys_logic_ngcs_bypass_spk_i5d1d2_o5ch1_io44k1	4,27	11,9
sys_logic_ngcs_outputmode_spk_i2ch_o5ch1_io44k1	3,90	11,5

Tabela 5.1 Utrošak procesorskog vremena i trajanje obrade virtualizacije za različite testne tokove (izvršavanje u simulatorskom režimu rada)

Podaci o potrošnji memorije, odnosno broju zauzetih lokacija, mogu se dobiti iz datoteke sa ekstenzijom .MAP koja se dobija prilikom generisanja izvršne datoteke. U tabeli 5.2 prikazana je potrošnja memorije korišćenje platforme, podeljena na potrošenu programsku memoriju i memoriju za podatke.

X memorija	Y memorija	L memorija	P memorija
292	41	1549	1208

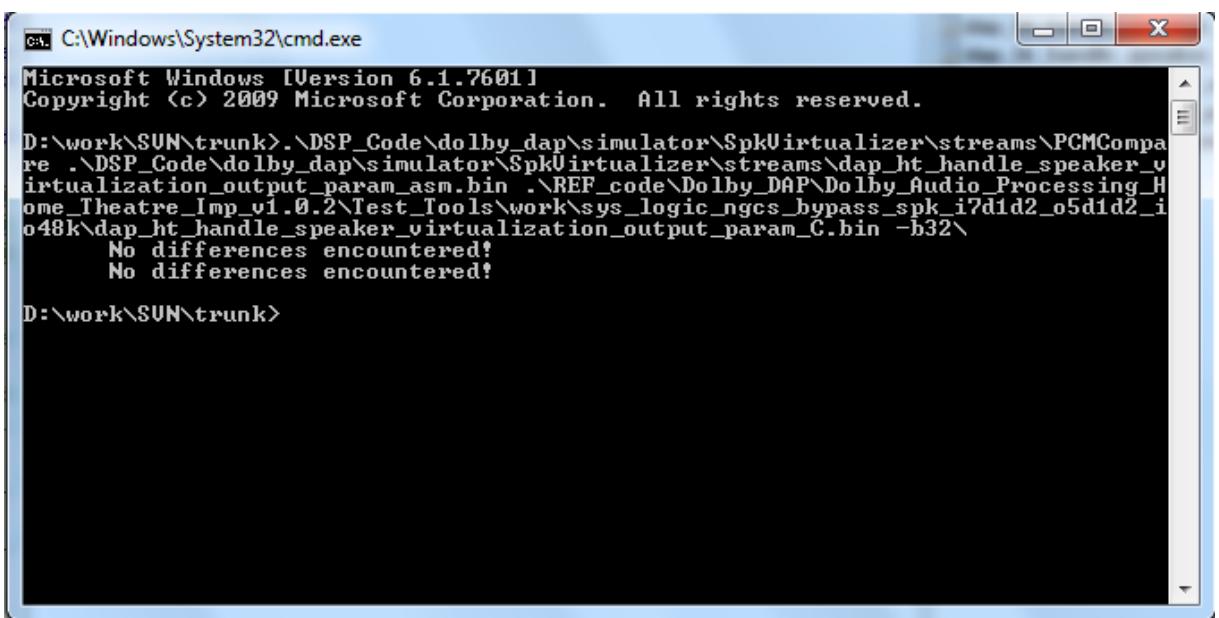
Tabela 5.2 Potrošnja memorije (broj reči)

5.2 Ručni bit-identični testovi

Ovaj vid testiranja podrazumeva poređenje na nivou bita izlaza generisanim pokretanjem referentnog programa sa izlazom generisanim nakon završetka asemblerorskog projekta. Za potrebe ovakvog testiranja korišćeni su alat PCMCompare.exe i Total Commander. Kako se i u samom referentnom programu koristi aritmetika nepokretnog zareza,

očekivan rezultat testiranja je potpuna identičnost datoteka na nivou bita. Kao što je i očekivano, pokazuje se identičnost izlaznih datoteka iz asemblerorskog projekta i iz referentnog programa.

Na Slici 5.1 može da se vidi izlaz jednog od ručno rađenih testova. Pošto je tokom izrade projekta i testiranja u međukoracima primećeno da se greške javljaju i nakon određenog vremena izvšavanja programa, kako bi se sa sigurnošću utvrdila ispravnost asemblerorskog projekta, svaki od tokova je testiran do kraja. Korišćeni su različiti testni slučajevi, kako bi se proverili svi delovi koda u projektu. Izlazne datoteke sadrže podatke iz bafera sa kanalima koji je prisutan nakon obrade, na izlazu. Na taj način je proverena ispravnost dve obrade u algoritmu. Prvo, podešavanje informacija o kanalima u baferu nakon postavljanja formata bafera na 5.1 ili 5.1.2 konfiguraciju, tako što se uporedi izlazni broj kanala. Drugo, virtualizacija sadržaja kanala, izlistavanjem i upoređivanjem svih kanala prisutnih na izlazu.



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

D:\work\SUN\trunk>.\DSP_Code\dolby_dap\simulator\SpkVirtualizer\streams\PCMCompare .\DSP_Code\dolby_dap\simulator\SpkVirtualizer\streams\dap_ht_handle_speaker_virtualization_output_param_asm.bin .\REF_code\Dolby_DAP\Dolby_Audio_Processing_Home_Theatre_Imp_v1.0.2\Test_Tools\work\sys_logic_ngcs_bypass_spk_i7d1d2_o5d1d2_i048k\dap_ht_handle_speaker_virtualization_output_param_G.bin -h32<
No differences encountered!
No differences encountered!

D:\work\SUN\trunk>
```

Slika 5.1 Ručno poređenje izlaza referentnog i simulatorskog projekta

Kako nije vršeno ispitivanje modula u okviru završnog projekta niti su rađeni slušni testovi, ovaj vid provere upoređivanjem izlaznih kanala iz asemblerorskog i referentnog koda je bio najbolji i najprecizniji način da se utvrdi ispravnost implementiranog rešenja

6. Zaključak

U ovom radu je implementiran DSP modul za virtualizaciju surround zvučnika na procesoru CS49844 kompanije Cirrus Logic.

Izrada projekta je podeljena u nekoliko faza. Prva faza je bila proučavanje referentnog modela i razumevanje algoritma. Ovaj deo izrade projekta je rađen u razvojnog okruženju Visual Studio 2010. Pored proučavanja algoritma, poklonjeno je dosta pažnje i korišćenim strukturama podataka.

Sledeći korak je bio implementacija funkcija u asemblerskom jeziku, prateći algoritam u okviru referentnog projekta. Ovaj proces je vršen u okviru razvojnog okruženja CLIDE. CLIDE poseduje veliki broj korisnih funkcionalnosti, koje su omogućile stalan pregled stanja u registrima, memoriji i postepeno kontrolisano izvršavanje. Kontrolisanim izvršavanjem je tokom ove faze vršena stalna provera ispravnosti napisanih funkcija, poređenjem sa izlazima u referentnom projektu.

Kada je dobijen ispravan modul za virtualizaciju, napisan u asemblerskom jeziku, bilo je potrebno isti integrisati u projekat sa ostalim modulima. Priprema za ovu fazu je urađena već pri pisanju asemblerskog koda, jer su neke od korišćenih struktura pripremljene pre poziva modula za virtualizaciju.

Nakon integracije modula u projekat, uviđeno je da je potrošnja memorije u granicama resursa koje poseduje platforma. Iz tog razloga je bilo moguće preći na računanje preciznog utroška memorije i miliona instrukcija po sekundi. Merenjem utroška resursa različitih testnih tokova u simulatorskom režimu rada je primećena razlika u potrošenom procesorskom vremenu i trajanju obrade pri virtualizaciji tokova sa različitim početnim i krajnjim formatima. Primećuje se da je obrada testnih tokova sa izlaznim formatom 5.1.2 zahtevnija,

što objašnjava činjenica da u tom slučaju, za razliku od izlaznog formata 5.1, se virtualizuju ne samo levi i desni surround kanali nego i levi i desni gornji surround kanali. Početno podešavanje izlaznog formata testnog toka takođe utiče na utrošak procesorskog vremena i trajanje obrade, i na osnovu izmerenih vrednosti se može primeti da je najzahtevnija obrada virtualizacija toka sa formatom 7.1.2 u izlaz formata 5.1.2.

Algoritmi za virtualizaciju surround zvučnika napreduju konstantno. Rešenje prikazano u okviru ovog projekta koristi neke od najnovijih i najefikasnijih metoda simuliranja surround zvuka. Iako utrošak resursa omogućava efikasno izvršavanje algoritma na DSP platformi, moguće je unapređenje postojeće implementacije. To bi se prvenstveno moglo postići kroz veći stepen paralelizacije instrukcija i uštedom memorije podataka, korišćenjem akumulatora i registara. Kao posledica bi se javila dosta lošija preglednost koda i nedostatak modularnosti, tako da je kao optimalan kod određen završni modul u implementaciji. Sva druga unapređenja postojećeg rešenja bi morala biti sprovedena još u toku izrade referentnog koda, kroz izmene samog algoritma. Ovaj postupak zahteva detaljno poznavanje svake od metoda virtualizacije i mogućih načina njihovog unapređivanja.

Algoritam implementiran u ovom projektu nije jednostavno proveriti. Poređenja na nivou bita svih testnih tokova su prošli očekivano, bez postojanja razlike među njima. Ovim metodom provere je verifikovano da je izlazni broj kanala ispravan i da je obrađeni sadržaj virtualizovanih kanala podudaran sa izlazom iz referentnog projekta. Slušni testovi nisu rađeni, pa nije bilo moguće verifikovati ispravnost projekta i na taj način. Međutim, za razliku od obrada vršenih u drugim modulima, za slušnu proveru virtualizatora surround zvučnika potrebna je odgovarajuća oprema i okruženje, tako da se uobičajenim slušnim proverama ne bi moglo utvrditi da li je postignut efekat surround zvuka.

7. Literatura

- [1] V. Kovačević, M. Popović, M. Temerinac, N. Teslić: „*Arhitekture i algoritmi digitalnih signal procesora I*“, FTN izdavaštvo, Novi Sad, 2005
- [2] „*Arhitekture i algoritmi digitalnih signal procesora – Zbirka zadataka i laboratorijski praktikum*“, u pripremi
- [3] Alan Kraemer: *Two Speakers Are Better Than 5.1*, IEEE Spectrum, 2001
- [4] Circuits Today, <http://www.circuitstoday.com/multi-channel-surround-sound-systems>, 7.07.2016
- [5] Virtual acoustics and audio engineering, *Optimal Source Distribution*, <http://resource.isvr.soton.ac.uk/FDAG/VAP/html/osd.html>, 7.07.2016
- [6] „*CS498xx_Datasheet*“, Cirrus Logic, Inc., 2013
- [7] M. Djukić, N. Četić, J. Kovačević, M. Popović, „*A C compiler based methodology for implementing audio DSP applications on a class of Embedded Systems*“, IEEE International Symposium on Consumer Electronics, 2008