



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ



Никола Андрић

Имплементација ДСП модула за управљање нискофреквентним садржајем

ДИПЛОМСКИ РАД
- Основне академске студије -

Нови Сад, 2015



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:	Никола Андрић		
Ментор, МН:	Доц. др Јелена Ковачевић		
Наслов рада, НР:	Имплементација ДСП модула за управљање нискофреквентним садржајем		
Језик публикације, ЈП:	Српски / латиница		
Језик извода, ЈИ:	Српски		
Земља публиковања, ЗП:	Република Србија		
Уже географско подручје, УГП:	Војводина		
Година, ГО:	2015		
Издавач, ИЗ:	Ауторски репримт		
Место и адреса, МА:	Нови Сад; трг Доситеја Обрадовића 6		
Физички опис рада, ФО: (поглавља/страница/цитата/табела/ слика/графика/прилога)	7/29/0/3/16/0/0		
Научна област, НО:	Електротехника и рачунарство		
Научна дисциплина, НД:	Рачунарска техника		
Предметно одредница/Кјучнеречи, ПО:	Дигитална обрада сигнала, Управљање нискофреквентним садржајем, Ефекти ниских фреквенција		
УДК			
Чувасе, 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 на ДСП процесору фирме Cirrus Logic.		
Датум приhvатања теме, ДП:			
Датум одбране, ДО:			
Чланови комисије, КО:	Председник:	Доц. др Небојша Ђевалица	
	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:	Nikola Andrić	
Mentor, MN:	PhD Jelena Kovačević	
Title, TI:	Implementation of Bass Manager DSP Module	
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:	2015	
Publisher, PB:	Author's reprint	
Publication place, PP:	Novi Sad, Dositeja Obradovica sq. 6	
Physical description, PD: (chapters/pages/ref./tables/pictures/graphs/appendices)	7/29/0/3/16/0/0	
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 solution for DSP Bass Manager module for Cirrus Logic digital signal processor.	
Accepted by the Scientific Board on, ASB:		
Defended on, DE:		
Defended Board, DB:	President:	PhD Nebojša Pjevalica
	Member:	PhD Ivan Kaštelan
	Member, Mentor:	PhD Jelena Kovačević
		Menthor's sign

Zahvalnost

Zahvaljujem se mentorima i članovima RT-RK DSP grupe na stručnoj pomoći tokom izrade rada.

Zahvaljujem se porodici i prijateljima na podršci tokom studiranja.

SADRŽAJ

1.	Uvod.....	1
2.	Teorijske osnove	2
2.1	LFE kanal i njegove karakteristike.....	2
2.2	Upravljanje niskofrekventnim sadržajem.....	4
2.3	Digitalna obrada signala i digitalni signal procesori	6
2.4	Digitalni filtri.....	7
2.5	Osnovne karakteristike procesora CS49834	7
2.6	Razvojno okruženje CLIDE i programsko okruženje Cirrus Logic Framework	9
3.	Koncept rešenja.....	13
3.1	Kratak opis algoritma	13
3.2	Tok razvoja implementiranog rešenja	14
3.2.1	Faza 1	15
3.2.2	Faza 2	15
3.2.3	Faza 3	15
4.	Implementacija.....	16
4.1	Faza 1	16
4.2	Faza 2	16
4.2.1	Datoteka BassExtractOsInterface	17
4.2.2	Datoteka BassExtractProcessing	17
4.2.3	Datoteka BassExtractFunc	18
4.2.4	Datoteke BassExtractCoefs i BassExtractDataVars	18
4.3	Faza 3	18
4.4	Iskorišćenost resursa ciljne platforme	19

5.	Ispitivanje.....	20
5.1	Ručni bit-identični testovi	23
5.2	Automatski bit-identični testovi	24
5.3	Slušni testovi i analiziranje spektralnih karakteristika.....	26
6.	Zaključak	27
7.	Literatura.....	29

SPISAK SLIKA

Slika 2.1 Primer 5.1 i 7.1 višekanalnih audio sistema	2
Slika 2.2 Primer sistema bez sabvufer izlaza i sistema sa sabvufer izlazom	5
Slika 2.3 Primer sistema sa ograničavačem	6
Slika 2.4 Blok dijagram procesora CS98434	8
Slika 2.5 Blok dijagram Crystal 32 DSP jezgra	9
Slika 2.6 Izgled prozora u CLIDE okruženju.....	10
Slika 2.7 Blok dijagram sprege modula sa operativnim sistemom	11
Slika 3.1 Blok dijagram algoritma	14
Slika 5.1 Spektralna karakteristika LFE kanala multiton skog testnog vektora	21
Slika 5.2 Spektralna karakteristika visokofrekventnog filtra za različite vrednosti parametra Cutoff.....	22
Slika 5.3 Spektralna karakteristika za različite vrednosti parametra Bass Boost	22
Slika 5.4 Prikaz signala u vremenskom domenu za različite vrednosti parametra Phase Shift	23
Slika 5.5 Ručno poređenje referentnih i izlaza simulatorskog projekta.....	23
Slika 5.6 Ručno poređenje referentnih i izlaza sa ciljne platforme	24
Slika 5.7 Ručno poređenje izlaza sa ciljne platforme i izlaza simulatorskog projekta	24
Slika 5.8 Deo tekstualne datoteke u kojoj su prikazani rezultati testiranja.....	25

SPISAK TABELA

Tabela 2.1 Raspoloživa memorija procesora CS49834	8
Tabela 4.1 Prikaz potrošnje resursa ciljne platforme	19
Tabela 5.1 Sadržaj testnog vektora multi_tone.wav	21

SKRAĆENICE

DSP	- <i>Digital Signal Processing</i> , Digitalna obrada signala
LFE	- <i>Low-Frequency Effects</i> , Efekti niskih frekvencija
MAC	- <i>Multiply And Accumulate</i>
AVR	- <i>Audio-Video Reciver</i> , Audio video prijemnik
MIPS	- <i>Million Instruction Per Second</i> , Milion instrukcija u sekundi
CLIDE	- <i>Cirrus Logic Integrated Development Enviroment</i> , Cirrus Logic integrисано razvojno okruženje
IIR	- <i>Infinite Impulse Response</i> , Beskonačan impulsni odziv
FIR	- <i>Finite Impulse Response</i> , Konačan impulsni odziv
OS	- Operating System, Operativni sistem
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

1. Uvod

Tema ovog rada je implementacija DSP modula za upravljanje niskofrekventnim sadržajem na DSP procesoru firme Cirrus Logic za kućne bioskope (eng. *Home-Theatere*).

Rad obuhvata upoznavanje sa osnovama audio DSP obrade i upoznavanje sa ciljnom platformom. Implementacija obuhvata prilagođavanje referentnog C koda, kao i realizaciju u asemblerskom jeziku ciljne DSP platforme. Cilj rada je upoznavanje sa pisanjem programske podrške u realnom vremenu za DSP aplikacije na ciljnoj platformi. Zadatak se oslanja na rad u programskim jezicima C i asemblerskom jeziku ciljne platforme.

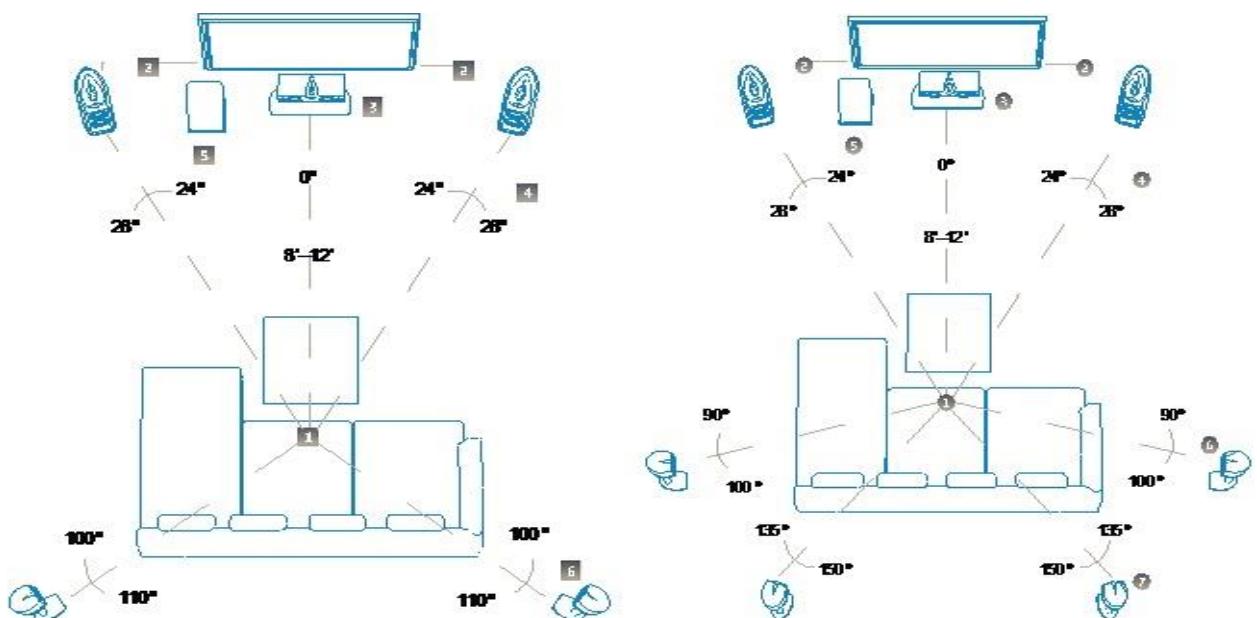
Rad se sastoji od sedam poglavlja. U drugom poglavlju su opisani teorijski pojmovi i koncepti na kojima se rad zasniva. Takođe, u okviru drugog poglavlja opisana je ciljna platforma. U trećem poglavlju predstavljen je koncept rešenja i opis toka implementacije. Četvrto poglavlje se odnosi na detalje implementacije. U petom poglavlju dat je opis postupka ispitivanja kao i rezultati ispitivanja za različite testne slučajeve. Šesto poglavlje sadrži zaključak, u kom je sažeto sve što je urađeno u okviru rada. Poslednje, sedmo poglavlje sadrži spisak literature korišćene u realizaciji rada.

2. Teorijske osnove

U ovom poglavlju dat je opis pojmove i koncepata na kojima se temelji ovaj rad. Obrazložena potreba za uvođenjem LFE kanala, i opisane osnovne ideje i zahtevi kontrole bas sadržaja (eng. *Bass Management*). Ukratko su navedene osobine i prednosti digitalne obrade signala, opisan je koncept digitalnih filtara. Na kraju, date su karakteristike ciljne platforme i razvojnog okruženja.

2.1 LFE kanal i njegove karakteristike

Moderni višekanalni audio sistemi se sastoje od nekoliko zvučnika punog opsega (eng. *Full range*), i jednog ili više zvučnika ograničenog opsega (eng. *Band-limited*). Za označavanje ovakvih sistema koristi se notacija x.x. Broj pre tačke označava broj kanala punog opsega, a broj posle tačke ukupan broj LFE kanala [6]. Na sledećoj slici dat je primer ovakvih sistema.



Slika 2.1 Primer 5.1 i 7.1 višekanalnih audio sistema

LFE kanal se prvobitno pojavio kod Dolby Stereo 70mm filmskih traka (1970-ih godina), a sredinom 1990-ih i 2000-ih je postao uobičajena pojava u kućnim bioskopima (eng. *Home Theater System*). Za razliku od osnovnih kanala, LFE kanal nosi jedino bas sadržaje, odnosno sadržaje sa frekvencijom nižom od 120Hz [5].

Sadržaj ovog kanala se najčešće šalje na zvučnik koji je namenski dizajniran za reprodukciju ovakvih sadržaja. Takav zvučnik naziva se sabvufer. Sabvuferi kod jeftinijih sistema obično preuzimaju reprodukciju niskofrekventnih sadržaja do oko 200 Hz zbog ograničenosti glavnih zvučnika, dok je kod profesionalnih sisteme ta granica niža i obično iznosi oko 80 Hz. Najčešće se sabvufer sastoji od jednog ili više zvučnika ugrađenih u kućište koje je izrađeno od drveta. Kućišta mogu biti dizajnirana na razne načine kako bi se postigao kompromis u vezi sa efikasnošću, propusnim opsegom, dimenzijama i cenom. Prvi sabvuferi su razvijeni u 1960-im za kućne stereo sisteme, a su postali naročito popularni sa pojmom filmova koji su sadržali niskofrekventne efekte namenjene za reprodukciju na sabvufer zvučnicima. Sabvufer može biti deo sistema zvučnika koji sadrži male zvučnike „satelite“, ili može biti ugrađen u isto kućište sa osnovnim zvučnicima (primer su takozvani „tower“ zvučnici). Sistemi zvučnika koji sadrže fizički odvojen sabvufer i „satelite“ su popularni kod kompleta zvučnika za kućne bioskope (eng. *Home Theater in a Box*).

LFE kanal nosi dodatne niskofrekventne sadržaje, koji mogu dopuniti niskofrekventni sadržaj glavnih kanala. Signal u LFE kanalu je kalibriran tokom snimanja i sposoban da priloži 10dB viši nivo zvučnog pritiska (eng. *Sound Pressure Level - SPL*) nego isti bas signal u nekom od osnovnih kanala. Ova osobina omogućila je filmskim stvaraocima da intenzivne bas signale poput zvuka eksplozija, zemljotresa ili lansiranja raketa smestite u LFE kanal [5].

Često se LFE kanal poistovećuje sa sabvuferom, međutim termini LFE kanal i sabvufer nisu zamenljivi. Moguće je da program sadrži LFE kanal, ali da dekoder ne generiše sabvufer izlaz pošto se sav bas sadržaj osnovnih kanala, kao i sadržaj LFE kanala može reprodukovati sa postojećim zvučnicima. Moguće je i drugi slučaj, da emitovani program ne sadrži LFE kanal, ali da dekoder generiše sabvufer izlaz pošto pojedini ili čak svi zvučnici u sistemu nisu sposobni da reprodukuju bas sadržaj. Razlika između LFE kanala i sabvufer zvučnika je ta što LFE prenosi dodatne niskofrekventne sadržaje emitovanog programa, a sabvufer zvučnik predstavlja način, odnosno komponentu na kojoj se bas sadržaj reprodukuje.

Kada je reč o sabvufer zvučniku, treba spomenuti njegovu sposobnost da reprodukuje bas sadržaj bilo kog kanala u sistemu. Izbor kanala čiji će bas sadržaj biti reprodukovani na sabvuferu vrši kontroler bas sadržaja (eng. *Bass Management*). Na primer, pored bas sadržaja iz LFE kanala, sabvufer može reprodukovati i bas sadržaj iz centralnog i okružujućih (eng. *surround*)

kanala, u slučaju da njima korespondentni zvučnici nisu u stanju da adekvatno reprodukuju bas frekvencije.

2.2 Upravljanje niskofrekventnim sadržajem

Termin upravljanje niskofrekventnim sadržajem (eng. *Bass Management*) pojavio se sredinom devedesetih godina prošlog veka. Ovaj termin je iskorišćen da opiše niz kompleksnih i naprednih funkcija u audio-video prijemnicima (eng. *Audio Video Recivers - AVR*), koje pomažu korisniku tokom konfigurisanja audio-video sistema sa akcentom na što bolju reprodukciju bas sadržaja. Iako je ovaj koncept nastao u 1980-im godinama, prava revolucija dogodila se kada su kompanije Dolby Digital i DTS izašle na tržište sa 5.1 kanalnim digitalnim formatima. Oba formata su podrazumevala postojanje, dodatnog kanala nazvanog LFE (eng. *Low Frequency Effects*), koji bi trebalo da prenosi niskofrekventne zvučne efekte visokog intenziteta. Koncept kontrole bas sadržaja temelji se na postojanju:

- sistema za određivanje frekvencijskog opsega svakog zvučnika u sistemu,
- frekvencijski prilagodljivih filtara za svaki zvučnik u sistemu, kako bi se obezbedilo da se na zvučnik šalje samo deo spektra signala koji je ovaj u stanju da reprodukuje,
- ograničavača amplitude bas sadržaja, kako bi se sprečilo preopterećenje zvučnika,
- složenog sistema za preusmeravanje bas sadržaja iz svih kanala
- sistema za procenu odziva prostorije, i eventualnu korekciju pomenutih filtara

Međutim, ovaj pristup se ne koristi u komercijalnim uređajima jer i manje sofisticirana rešenja daju zadovoljavajuće karakteristike.

Kod svih modernih audio-video uređaja mora da postoji određeni oblik upravljanja niskofrekventnim sadržajem, koji uključuje visokopropusne filtre za svaki od glavnih kanala (levi, desni, centralni, levi okružujući, desni okružujući, zadnji levi i zadnji desni), kao i odgovarajući niskopropusni filter za LFE kanal. Ovo je neophodno da bi AVR uređaji mogli da podrže širok spektar modela zvučnika koji su dostupni na tržištu. Ovi filtri treba da budu konfigurisani na osnovu dva faktora, a to su :

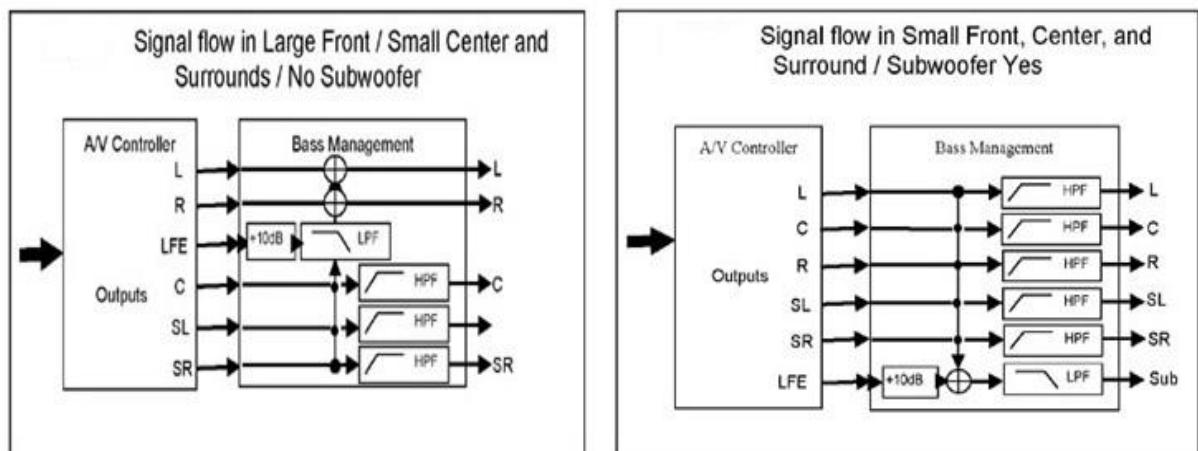
- Sposobnost glavnih zvučnika da reprodukuju niskofrekventne sadržaje
- Oblik prostorije u kojoj će sistem zvučnika biti postavljen,

U idealnom slučaju, korisnik bi mogao sam da podesi filter na idealnu graničnu frekvenciju za svaki zvučnik posebno. Međutim, najčešće je vrednost te frekvencije zajednička za sve zvučnike u sistemu. Iako ideja o frekvencijski prilagodljivim filtrima zvuči kao bolja, filtri sa predefinisanom graničnom frekvencijom postižu dovoljno dobre rezultate. Svaka prostorija može se posmatrati kao rezonantni sistem, u kom postoji beskonačan niz rezonantnih frekvencija. Sa

porastom frekvencije gustina rezonanci se povećava. U oblasti od najniže rezonantne frekvencije pa do granice gde gustina rezonanci postaje dovoljno velika, u odzivu prostorije mogu se konstatovati efekti koji nastaju formiranjem stoećih talasa. Ta zona na frekvencijskoj osi naziva se kritični opseg. U ovoj oblasti javljaju se značajne nelinearnosti frekvencijskog odziva. Zona kritičnog opsega na frekvencijskoj osi zavisi od dimenzija prostorije. U malim prostorijama, kao što su sobe u stanovima, najniža rezonanca se nalazi u opsegu između 50 i 100 Hz, pa je čitav kritični opseg u području gde se nalaze bitne komponente korisnog signala. U zavisnosti od pozicije izvora i prijemnika u prostoru neregularnosti odziva su drugačije. Izborom pozicije izvora teži se smanjenju uticaja ovih neregularnosti. Treba obratiti pažnju na to da je signal koji dolazi na sabvufer ograničenog opsega negde između 80 i 120 Hz. Sa porastom vrednosti granične frekvencije, povećava se mogućnost lokalizacije sabvufera. Pošto LFE kanal može sadržati komponente oko 120 Hz, preporučeno je da granična frekvencija bude 120 Hz. Međutim, smanjenjem vrednosti granične frekvencije (najčešće na 80 Hz) postiže se veća fleksibilnost pri pozicioniranju sabvufera [7].

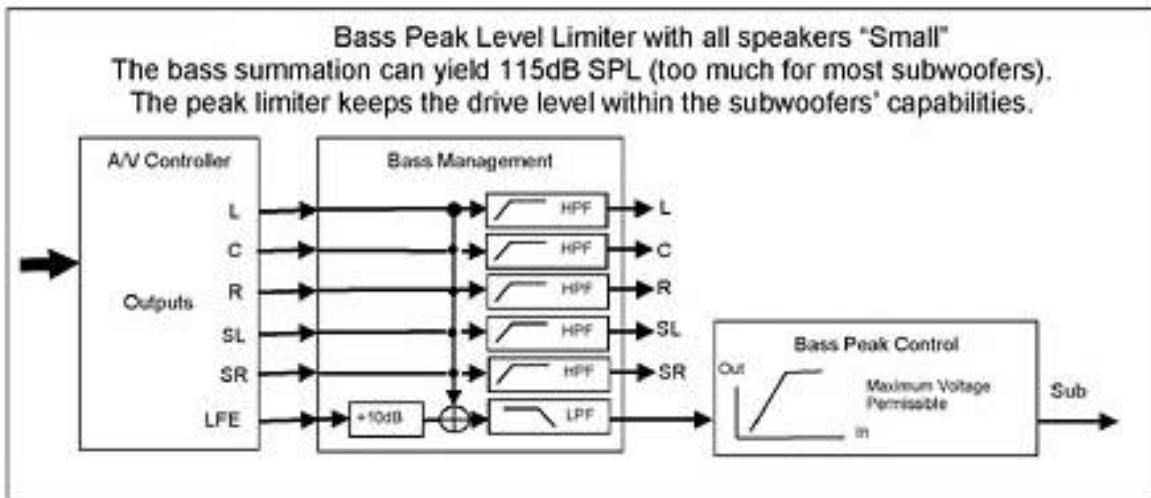
Sledeće što treba razmatrati kod upravljanja bas sadržajem jeste preusmerenje bas sadržaja. U idealnim sistemima bas sadržaj iz ma kog kanala bi bilo moguće preusmeriti u ma koji drugi kanal, na bilo kom nivou i bilo kojoj frekvenciji.

Neki od audio-video sistema sadrže sabvufer, a neki ne. Primer i jednog i drugog sistema su predstavljeni na sledećim slikama [4]. Na levoj strani slike prikazan je sistem bez sabvufera, u tom slučaju bas sadržaj iz centralnog, okružujućih (eng. *Surround*) i LFE kanala se šalje u prednje (eng. *Front*) zvučnike. U drugom slučaju, prikazanom na desnoj strani slike, sav bas sadržaj iz prednjih, centralnog, okružujućih kanala zajedno sa sadržajem LFE kanala se šalje u sabvufer. Ponekad sistem nudi opciju da se bas sadržaj iz „malih“ zvučnika, koji ne mogu reprodukovati bas, šalje i na „velike“ zvučnike i na sabvufer. To nije poželjno, iz razloga što može doći do interferencije.



Slika 2.2 Primer sistema bez sabvufer izlaza i sistema sa sabvufer izlazom

Na kraju kada su visokopropusne i niskopropusni filtri konfigurisani, i kompletan bas sadržaj raspoređen na odgovarajuće zvučnike, u sistem se uključuje ograničavač nivoa bas sadržaja. Uloga ograničavača je da spreči previsok nivo izlaznog napona koji bi mogao preopteretiti zvučnike ili sabvufer. Jedan takav sistem je prikazan na sledećoj slici.



Slika 2.3 Primer sistema sa ograničavačem

2.3 Digitalna obrada signala i digitalni signal procesori

Uopšteno, digitalna obrada signala podrazumeva pretvaranje fizičke pojave (zvuka, slike) u električne signale i njihovu diskretizaciju radi obrade na procesoru. Brz razvoj poluprovodničke tehnologije omogućio je integraciju moćnih i efikasnih procesora u uređaje koji se koriste u gotovo svim oblastima ljudskog života. Digitalna obrada signala se temelji na matematici, algoritmima i tehnikama za obradu koji su pogodni za programiranje na procesoru. Glavne oblasti digitalne obrade signala su obrada audio signala, audio kompresija, digitalna obrada slike, digitalna obrada video signala, video kompresija, algoritmi za obradu signala govora, prepoznavanje govora, radarski sistemi, seizmologija, biomedicina i drugo.

U zavisnosti od složenosti i zahteva algoritma (aplikacije), digitalnu obradu signala moguće je vršiti i na procesorima opšte namene, međutim digitalni signal procesori su obično jeftiniji i efikasniji, a samim tim pogodniji za ugradnju u uređaje potrošačke elektronike. Osnovne karakteristike digitalnih signal procesora su postojanje specijalizovanih instrukcija karakterističnih za obradu signala (MAC), Harvard arhitektura (odvojena programska i memorija za podatke), podrška za hardverske petlje, specijalizovani adresni režimi i druge [1]. Digitalna obrada signala najčešće podrazumeva rad u realnom vremenu.

2.4 Digitalni filtri

Digitalni filtri [2] su najčešće primenjivan sistem u digitalnoj obradi signala. Osnovna funkcija diskretnog filtra je frekvencijski selektivna modifikacija ulaznog signala. Najčešće je reč o propuštanju spektralnih komponenti iz jednog opsega (propusni opseg) i potiskivanju ostalih spektralnih komponenti (nepropusni opseg). Postoje tri osnovna tipa filtara, a to su: nisko-frekventni filtri (NF, eng. *Low Pass-LPF*), visoko-frekventni (VF, eng. *High Pass- HPF*) i pojasni filtri (PF, eng. *Band Pass-BPF*).

Digitalni filtri su linearni, vremenski invarijantni sistemi (LVIS) pa se generalno mogu opisati pomoću jednačine koja povezuje odbirke izlaznog signala $y(n)$ sa odbircima ulaznog signala $x(n)$:

$$y(n) = \sum_{k=0}^{L-1} a(k) * x(n - k) - \sum_{k=1}^M b(k) * y(n - k)$$

Parametri filtra (koeficijenti) $a(k)$ opisuju direktni deo koji je zavisan od ulaznih odbiraka, dok koeficijenti $b(k)$ opisuju rekurzivni deo koji je zavisan od prethodnih izlaznih odbiraka. Prenosna karakteristika $H(f)$, preko z -transformacije $H_z(z)$, se definiše kao Furijeova transformacija diskretnog impulsnog odziva filtra $h(n)$:

$$x(n) = \delta(n) \Rightarrow y(n) = h(n)$$

$$H_z(z) = \sum_n h(n) * e^{-2j\pi f n} = \frac{\sum_{k=0}^L a(k) * z^{-1}}{1 + \sum_{k=1}^M b(k) * z^{-1}} \Leftrightarrow H(f) = H_z(z^{2j\pi f})$$

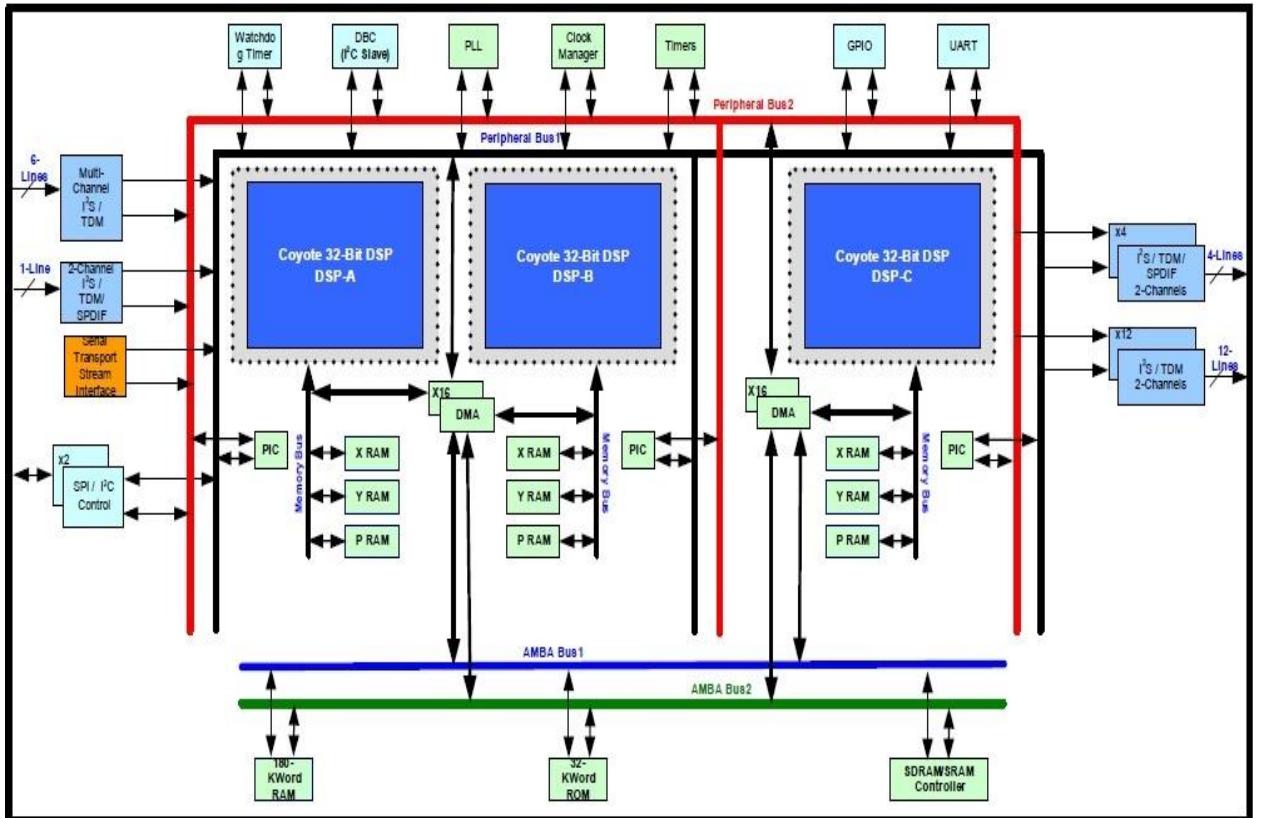
Za filtre sa konačnim impulsnim odzivom se koristi skraćenica FIR (eng. *Finite Impulse Response*) i oni nemaju rekurzivni deo. Filtri sa beskonačnim impulsnim odzivom, IIR (eng. *Infinite Impulse Response*) imaju i rekurzivni i direktni deo. Generalno, prednosti FIR filtara su garantovana stabilnost i jednostavno ispunjavanje zahteva linearnosti fazne karakteristike, dok su prednosti IIR filtera manji broj potrebnih operacija za postizanje istog rezultata.

Implementacija filtara zahteva da broj koeficijenata (L i M) bude konačan. Kauzalnost i konačnost realnih diskretnih sistema isključuju mogućnost postizanja idealne karakteristike filtra, međutim mogu se postići zadovoljavajuće aproksimacije.

2.5 Osnovne karakteristike procesora CS49834

Digitalni signal procesori iz familije CS49XXX kompanije Cirrus Logic namenjeni su obradi audio signala u uređajima potrošačke elektronike. Procesori iz ove familije zasnovani su na Crystal arhitekturi DSP jezgra, a razlike između pojedinih modela se odnose na broj jezgara, radni takt i količinu interne memorije.

CS49834 [8] je trojezgarni procesor sa radnim taktom od 300MHz, što ga čini jednim od najmoćnijih procesora iz Crystal familije. Zasnovan je na unapređenoj Harvard [1] arhitekturi koja podrazumeva odvojenu memoriju za podatke i memoriju za instrukcije. Dodatno, memorija za smeštanje podataka je podeljena u dva segmenta. Svako jezgro predstavlja tridesetdvobitni programabilan DSP koji radi sa aritmetikom u fiksnom zarezu. Na sledećoj slici prikazan je funkcionalni blok dijagram CS49834 procesora, a u tabeli su dati podaci o memoriji.



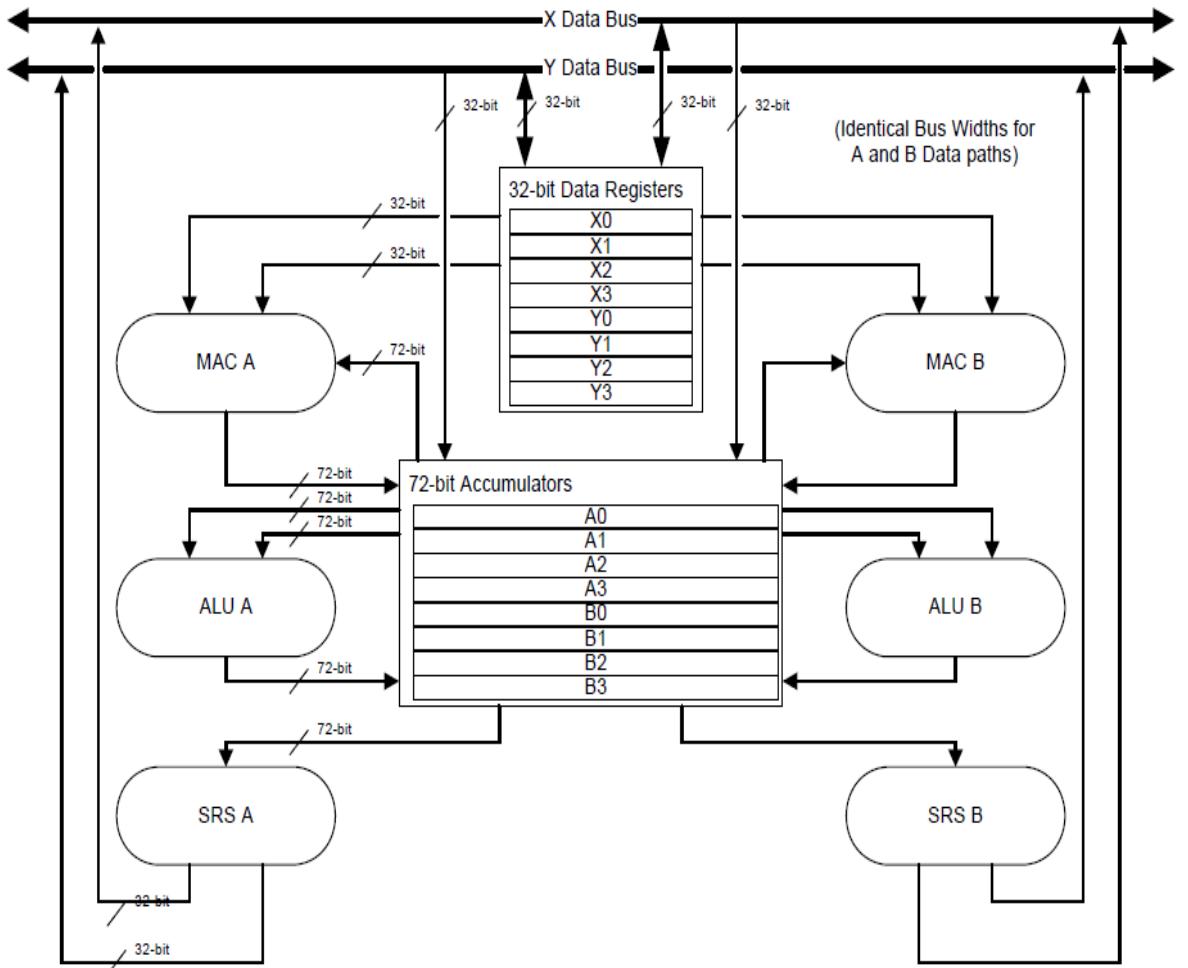
Slika 2.4 Blok dijagram procesora CS98434

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

Tabela 2.1 Raspoloživa memorija procesora CS49834

Svako jezgro sadrži jedinicu za kontrolu toka programa, paralelne jedinice za generisanje adresa i paralelne putanje toka podataka (A i B). Jedinice za generisanje adresa (eng. *Address Generation Unit - AGU*) raspolažu sa dvanaest indeksnih registara (i0-i11) dužine 16 bita, kao i dvanaest njima odgovarajućih modulo registara (mn0-mn11). Upotreboom ovih registara u sprezi obezbeđuju se različiti adresni režimi. Jezgro sadrži osam 72-bitnih akumulatorskih (a0-a3, b0-b3) registara namenjenih za čuvanje krajnjih i među rezultata MAC instrukcije i drugih aritmetičkih operacija. Akumulatori se sastoje od tri pod registra označenih kao Guard (8 bita), High (32 bita) i Low (32 bita), pri čemu se svakom delu može pristupiti zasebno. Pored

akumulatorskih, jezgro sadrži i osam 32-bitnih registara opšte namene (x0-x3, y0-y3). Kao što je predstavljeno na sledećoj slici svaka putanja toka podataka sadrži MAC (eng. *Multiply-Accumulate*), ALU (eng. *Arithmetic and Logical Unit*) i SRS (eng. *Shifter/Rounder/Saturator*) jedinicu.

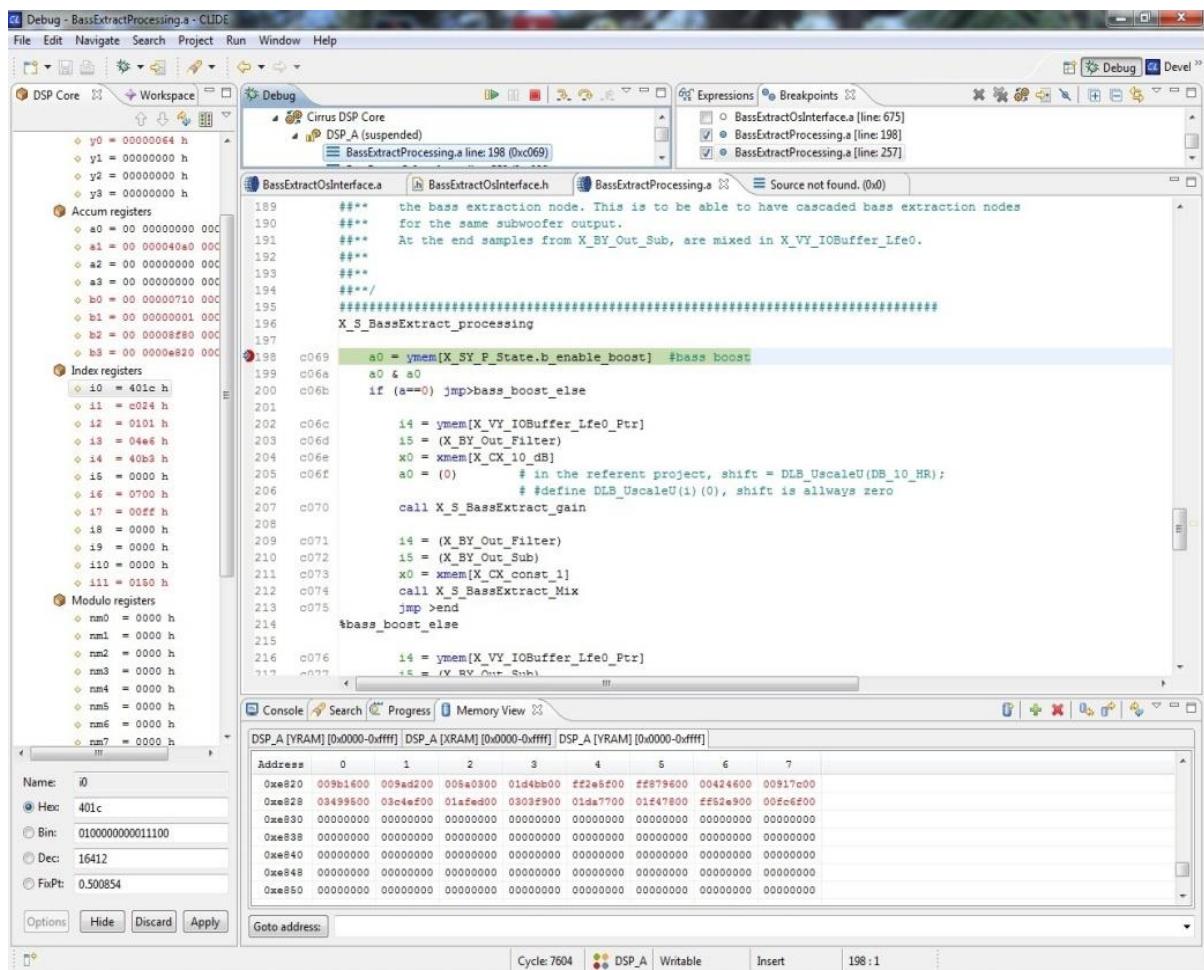


Slika 2.5 Blok dijagram Crystal 32 DSP jezgra

2.6 Razvojno okruženje CLIDE i programsko okruženje Cirrus Logic Framework

Za razvoj asemblerorskog koda u ovom radu, korišćeno je integrisano razvojno okruženje namenjeno digitalnim signal procesorima iz familije Crystal. Ovo namensko razvojno okruženje nosi naziv CLIDE (*Cirrus Logic Integrated Development Environment* [3]), i zasniva se na Eclipse razvojnom okruženju. Razvoj programske podrške za DSP platforme u velikoj meri se olakšava upotrebom CLIDE-a ili sličnih alata, iz razloga što oni omogućuju kontrolisano izvršavanje programa kao i opciju izvršavanja u simulatorском režimu. Kontrolisano izvršavanje često uključuje i opcije poput prikaza memorije i prikaza registara, što omogućuje efikasnu detekciju i otklanjanje grešaka. Simulatorски režim je izuzetno važan kod razvoja algoritama jer

ne obavezuje korišćenje potencijalno skupih razvojnih ploča. Na sledećoj slici dat je prikaz jednog prozora u CLIDE okruženju.



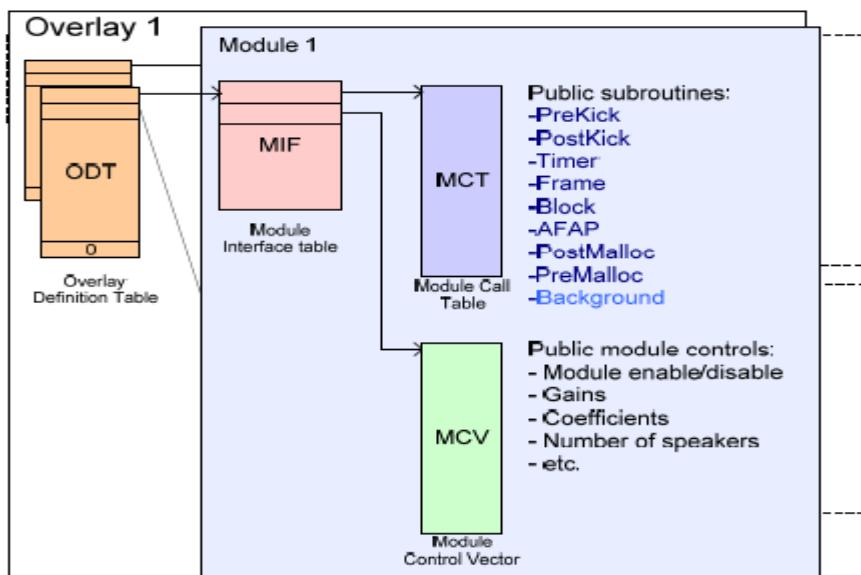
Slika 2.6 Izgled prozora u CLIDE okruženju

Sistemska programska podrška Cirrus Logic Framework [3], predstavlja programsko okruženje koje skraćuje vreme i uloženi rad za razvoj aplikacije. Jezgro programskog okruženja se sastoji od jednostavnog operativnog sistema koji ima ulogu raspoređivača za određen broj procesnih entiteta (modula). OS predstavlja monitorsku petlju koja poziva rutine odgovarajućih modula po unapred definisanom redosledu.

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 najznačajnije tabele su MCT (*Module Call Table*), i MCV (*Module Control Vector*). OS se povezuje sa modulima pomoću ODT tabele (*Overlay Definition Table*) koja sadrži pokazivače na MIF tabele svih učitanih modula.

Na sledećoj slici (Slika 2.7) može se videti sadržaj MCT i MCV tabela. 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 konfiguriranje modula od strane glavnog kontrolera uređaja (*host*). Ova tabela nema unapred definisani strukturi, već programer formuliše njen sadržaj i strukturu.



Slika 2.7 Blok dijagram sprege modula sa operativnim sistemom

U nastavku je dat kratak opis rutina definisanih u MCT tabeli.

Pre-Kickstart rutinu OS poziva samo nakon prijema inicijalizacione poruke i pre uspostavljanja komunikacije sa sistemskim kontrolerom. Ona omogućava inicijalizaciju modula, pre svega elemenata MCV tabele. Nakon što su izvršene *Pre-Kickstart* rutine svih modula, OS prelazi u stanje čekanja na *kickstart* poruku od kontrolera. Po prijemu ove poruke, nastavlja se sa pozivima *Post-Kickstart* rutina.

Post-Kickstart rutina se poziva nakon uspostave komunikacije sa kontrolerom. Drugim rečima ona omogućava obradu konfiguracionih poruka prosleđenih modulu od strane kontrolera. Nakon završetka *Post-Kickstart* rutine svih modula završena je faza inicijalizacije i OS prelazi na normalno izvršavanje ciklusa modula.

Pre-Malloc rutina se prvi put poziva nakon završetka inicijalizacionih rutina ukoliko je bilo koji modul u sistemu tokom inicijalizacije od OS-a zatražio inicijalizaciju dinamički dodeljene memorije. Po prijemu zahteva, OS poziva *Pre-Malloc* rutine aktivnih modula, unutar kojih moduli prosleđuju zahteve za dodelu memorije OS-u. Pri tome se navodi memorijska zona (X, Y ili L memorija) kao i veličina zahtevane memorije i pokazivač koji treba da se inicijalizuje.

Post-Malloc rutina se poziva nakon završetka alokacije dinamičke memorije. Ova rutina omogućava inicijalizaciju dinamički dodeljene memorije.

Timer rutina se poziva periodično svakih N milisekundi. Prvi put se poziva odmah nakon *Post-Kickstart* rutine i pre poziva *Frame* rutine. Ova rutina se može koristiti za proveru ulaznih konfiguracionih podataka modula.

Block i **Frame** su upravljane ulaznim tokom podataka. **Block** rutina se izvršava pri prijemu svakih 16 PCM odbiraka u U/I nizu, dok se *Frame* rutina izvršava na svakih N blokova, u zavisnosti od nivoa obrade i algoritma koji se izvršava.

AFAP rutina se poziva kod god se desi neki događaj u sistemu, uz uslov da ne prekida druge rutine istog prioriteta. AFAP je skraćenica od *As Fast As Possible*.

AFAP, **Timer**, **Frame** i **Block** čine takozvanu **Foreground Thread** (nit).

Background rutina se izvršava u pozadinskoj niti (**Background Thread**) i onima niži prioritet od **Foreground** niti. U slučaju dekodera unutar ove rutine se preuzimaju podaci iz ulaznog komprimovanog toka, obavlja se dekodovanje pri čemu će dekodovani PCM odbirci privremeno biti smešteni u lokalne memorijske nizove. Kasnije se korišćenjem AFAP rutine kopiraju u sistemski ulazno/izlazni memorijski niz. Kod algoritama završne obrade u *Block* rutini se kopira novi, neobrađeni blok od 16 PCM odbiraka iz ulazno/izlaznog memorijskog niza u lokalni memorijski niz nad kojim se potom vrši obrada. Istovremeno sa čuvanjem ulaznih odbiraka vrši se kopiranje prethodno obrađenih (izlaznih) odbiraka u ulazno/izlazni memorijski niz.

3. Koncept rešenja

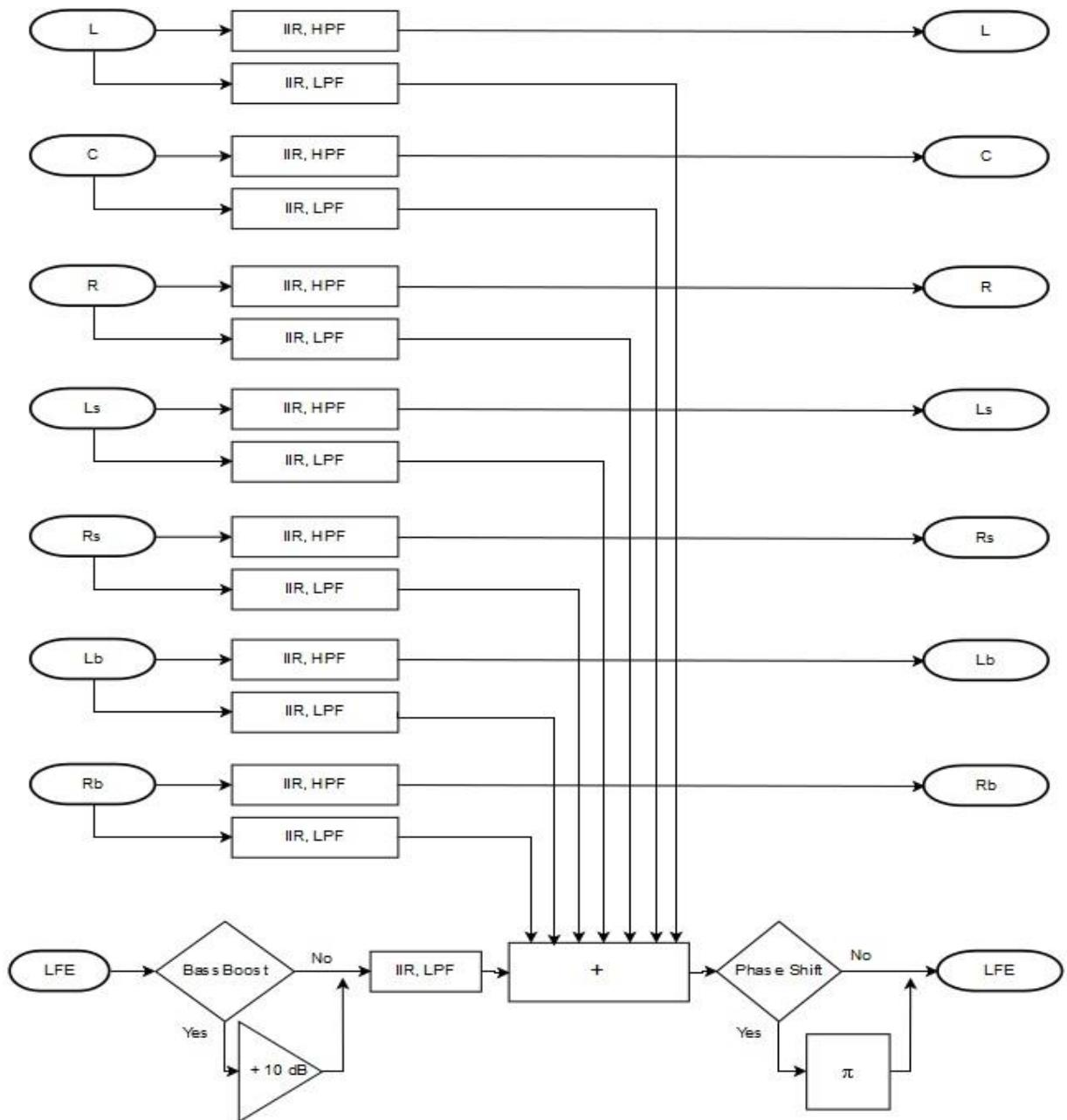
3.1 Kratak opis algoritma

Cilj ovog rada je implementacija DSP modula za upravljanje bas sadržajem. Algoritam u određenoj meri prati korake koji su navedeni u poglavlju 2.2. Početni korak podrazumeva konfigurisanje filtra na osnovu frekvencije odabiranja ulaznog toka podataka i željene granične frekvencije filtra. U ovoj implementaciji, granična frekvencija je konstantna za sve filtre. Naredni korak obuhvata filtriranje sadržaja svakog od kanala. Samo filtriranje se vrši pomoću IIR filtra drugog reda, odnosno kaskadne veze dva IIR filtra.

Sadržaj osnovnih kanala se filtrira visokopropusnim i niskopropusnim filtima. Pri tom se komponente sa frekvencijama višim od granične zadržavaju u osnovnim kanalima, dok se niskofrekventni sadržaji iz ovih kanala sumiraju sa niskofrekventnim sadržajem LFE kanala.

Sadržaj LFE kanala se filtrira niskopropusnim filtrom. Pre filtriranja sadržaja LFE kanala, može se uvećati za 10dB, zavisno od toga da li je parametar BassBoost uključen ili ne. Nakon filtriranja, u zavisnosti od vrednosti parametra PhaseShift, vrši se promena faze sadržaja LFE kanala za π (180°).

Upotrebom ovog algoritma na kraju se postiže da LFE kanal sadrži niskofrekventne komponente (do granične frekvencije), a ostali kanali frekvencije više od granične. Na sledećoj slici prikazan je blok dijagram ovog algoritma.



Slika 3.1 Blok dijagram algoritma

3.2 Tok razvoja implementiranoq rešenja

Razvoj aplikacija za ciljnu platformu predstavlja iterativni postupak. Svaka iteracija predstavljena je kao jedna faza razvoja. Prvu fazu predstavlja originalni, referentni kod. Naredne faze podrazumevaju implementiranje algoritma koji će se izvršavati na ciljnoj platformi, i imati funkcionalnost realizovanu u referentnom programu. One najčešće obuhvataju promenu radnog okruženja i programskog jezika u kom će se algoritam implementirati. Svaka naredna iteracija, odnosno faza, predstavlja korak bliže ka konačnom rešenju.

Tok realizacije ovog rada podeljen je u tri faze. U nastavku ovog poglavlja opisane su karakteristike pojedinih faza.

3.2.1 Faza 1

Početnu fazu u toku razvoju ovog DSP modula predstavlja referentni kod. U prvoj fazi, jedini zahtev koji se stavlja pred algoritam jeste ispravnost. Pošto se referentni kodovi najčešće razvijaju na računarima opšte namene, nije bitno voditi računa o efikasnom iskorišćenju memorije i procesorskog vremena, niti o aritmetici sa fiksnim zarezom, i drugim ograničenjima ciljnih platformi. Referentni kod je najčešće pisan u programskom jeziku C odnosno C++, ili u programskom paketu Matlab.

3.2.2 Faza 2

Druga faza u razvoju obuhvata implementaciju algoritma u asemblerском jeziku ciljne platforme. Ova faza podrazumeva rad u simulatorskom režimu. Na samom početku ove faze od izuzetnog značaja je napraviti dobru organizaciju resursa koje nudi ciljna platforma, ili imati barem okvirnu sliku o tome koliko resursa će biti iskorišćeno. Osnovni razlog za to je ograničenost resursa na ciljnoj platformi. Sledeći korak predstavlja samu implementaciju rešenja u asemblerском jeziku ciljne platforme, tokom ovog koraka akcenat se stavlja na ispravnost algoritma. U najvećem broju slučajeva kod generisan u ovoj fazi je izvršiv na ciljnoj platformi. Međutim, često se dešava da nije dovoljno prilagođen, odnosno da prevazilazi ograničenja ciljne platforme.

Zbog toga ova faza obično sadrži i dodatnu podfazu koja obuhvata optimizaciju asemblerског koda. Cilj ovog procesa je smanjenje korišćenih resursa (programske memorije, memorije za podatke i MIPS-a). Tehnike optimizacije se najčešće dele u sledeće grupe:

- Optimizacije koje koriste mogućnosti paralelnog izvršavanja više operacija procesora u okviru jedne instrukcijske reči i
- Optimizacije koje se postižu promenama u samom algoritmu

Krajnji rezultat ove faze je optimizovan asemblerski kod, prilagođen izvršavanju na ciljnoj platformi.

3.2.3 Faza 3

Kod generisan u prethodnoj fazi je prilagođen izvršenju na ciljnoj platformi, po pogledu iskorišćenja resursa kojima raspolaže platforma. Preostali koraci, kojima se završava ova faza, podrazumevaju određena konfiguraciona podešavanja koja zahteva ciljna platforma. Nakon završetka ove faze kod je izvršiv na ciljnoj platformi i spreman za ispitivanje i verifikaciju.

4. Implementacija

U ovom poglavlju su izneti detalji implementacije za svaku od faza opisanih u prethodnom poglavlju. Na kraju, tabelarno su prikazani detalji vezani za iskorišćenost resursa ciljne platforme.

4.1 Faza 1

Faza 1 predstavlja referentni kod koji je polazna osnova za implementaciju algoritma na ciljnoj platformi. Referentni kod može se podeliti u dve celine. Prva obuhvata module koji obezbeđuju pravilan rad sa ulaznom i izlaznom datotekom, implementirani su mehanizmi za otvaranje ulazne i izlazne datoteke, kao i mehanizmi za pravilno iščitavanje ulaznih odbiraka i njihov upis u izlaznu datoteku nakon obrade.

U okviru druge celine realizovani su moduli koji omogućuju obradu učitanih odbiraka na način koji je opisan u prethodnom poglavlju (3.1). Moduli iz druge celine preuzeti su iz jednog komercijalnog projekta.

4.2 Faza 2

Faza 2 podrazumeva promenu radnog okruženja i promenu programskog jezika. Referentni projekat bio je realizovan u programskom jeziku C, u radnom okruženju *Microsoft Visual Studio 2010*. Kod iz faze 2 implementiran je u asemblerskom jeziku ciljne platforme, u radnom okruženju CLIDE.

Prilikom razvoja u ovoj fazi korišćena je opcija CLIDE razvojnog okruženja za rad u simulatorskom režim. Ovaj režim kao ciljnu platformu koristi Cirrus DSP Core koji predstavlja simulator Crystal jezgra.

Modul je nazvan BassExtract, i sastoji se iz nekoliko celina (sve sadrže i .a i .h datoteke):

- BassExtractOsInterface
- BassExtractDataVars

- BassExtractProcessing
- BassExtractFunc

U nastavku je opisana uloga pojedinih datoteka u funkcionisanju modula.

4.2.1 Datoteka BassExtractOsInterface

U okviru datoteke realizovana je sprega ovog modula sa OS-om. Kao što je objašnjeno u poglavlju 2.6. Sprega se ostvaruje prozivanjem rutina iz MCT tabele. U okviru ove datoteke definisana je i MCV tabela, u kojoj su smeštene korisničke kontrole implementiranog modula. Pravi se i kopija MCV tabele čija se polja koriste tokom obrade.

U *Pre-Kickstart* rutini postavljaju se korisničke kontrole (polja MCV tabele) na podrazumevane vrednosti, a polja kopije se postavljaju na nulte vrednosti.

U okviru *Post-Kickstart* rutine anuliraju se vrednosti promenljivih koje se koriste kao kontrolne promenljive za označavanje uspešno izvršene inicijalizacije i uspešnog ažuriranja polja kopije MCV tabele.

Kao što je opisano, *Frame* rutina se poziva na svakih N blokova. U prvom pozivu ove rutine vrši se kopiranje korisničkih kontrola iz originalne tabele u kopiju. Po završetku kopiranja, označava se da je kopiranje izvršeno postavljanjem odgovarajuće kontrolne promenljive na jedinicu. Nakon toga vrši se inicijalizacija, koja podrazumeva konfigurisanje filtara, odnosno izbor koeficijenata i postavljanje odgovarajućih bafera koji modeluju stanja filtara. U svakom sledećem pozivu *Frame* rutine, neće se vršiti pomenuti koraci pošto su kontrolne promenljive postavljene na jedinicu. Međutim, vrši se provera da li je došlo do promene neke od korisničkih kontrola. Svaka promena praćena je ažuriranjem odgovarajuće promenljive u kopiji MCV tabele.

Block rutina poziva se pri prijemu svakih 16 PCM odbiraka ulaznog signala. U okviru ove rutine poziva se funkcija za obradu (X_S_BassExtract_processing).

Pošto su svi memorijski nizovi koji se koriste u okviru modula zauzeti statički, nije bilo potrebe za dodelu memorije u *PreMalloc* rutini, i kasnijeg inicijalizovanja dinamički zauzete memorije u *PostMalloc* rutini.

Kao što je pomenuto na početku poglavlja, radi se u simulatorskom režimu, što podrazumeva neophodno konfigurisanje simulatora pre pozivanja rutina iz MCT tabele. Konfigurisanje simulatora vrši se pozivom funkcije I_S_SimulatorInit.

4.2.2 Datoteka BassExtractProcessing

U okviru datoteke BassExtractProcessing realizovane su funkcije koje obezbeđuju ispravan rad algoritma:

- X_S_BassExtract_processing,
- X_S_BassExtract_iir1_df,
- X_S_BassExtract_iir2_lr,
- X_S_BassExtract_gain.

Glavna funkcija je X_S_BassExtract_processing, u kojoj je implementirana logika algoritma, u okviru nje se pozivaju ostale funkcije. U funkciji X_S_BassExtract_iir1_df implementirana je funkcionalnost IIR filtra. Funkcija X_S_BassExtract_iir2_lr modeluje kaskadnu formu IIR filtra drugog reda, što se realizuje pozivanjem funkcije X_S_BassExtract_iir1_df dva puta, naravno, za odgovarajuće parametre. X_S_BassExtract_gain izvršava množenje ulaznih odbiraka sa koeficijentom datim u parametru gain.

4.2.3 Datoteka BassExtractFunc

U ovoj datoteci realizovane su funkcije :

- X_S_BassExtract_Mix,
- X_S_BassExtract_State_Init,
- X_S_BassExtract_Clear,
- X_S_BassExtractCoef_Update

Funkcija X_S_BassExtract_Mix izvršava MAC instrukciju nad ulaznim podacima. Koristi se i u prethodnoj datoteci za potrebe prebacivanja sadržaja jednog bafera u drugi.

Funkcija X_S_BassExtract_State_Init se poziva u datoteci BassExtractOsInterface, u Frame rutini. Prilikom njenog izvršavanja poziva se funkcija X_S_BassExtract_Coeff_Update, koja obezbeđuje pravilno postavljanje koeficijenata nisko propusnih i visoko propusnih filtara, kao i X_S_BassExtract_Clear koja inicijalizuje početna stanja filtara za svaki kanal. Pored toga tokom izvršavanja X_S_BassExtract_State_Init postavljaju se parametri od kojih zavisi tok izvršavanja X_S_BassExtract_processing funkcije, a to su Bass Boost i Phase Shift.

4.2.4 Datoteke BassExtractCoefs i BassExtractDataVars

U okviru ovih datoteka statički je zauzeta memorija za sve promenljive (pokazivače, tabele, bafere) koje se koriste u modulu. U datoteci BassExtractCoefs statički je zauzeta memorija za koeficijente niskopropusnih i visokopropusnih filtara. Koeficijenti su organizovani u tabele, razvrstane po frekvenciji odabiranja ulaznog signala (32kHz-192kHz), a unutar tabela koeficijenti su složeni po graničnim frekvencijama od 45Hz od 200Hz. U datoteci BassExtractDataVars definisane su ostale globalne promenljive koje se koriste u modulu.

4.3 Faza 3

Za potrebe ove faze započinje se novi projekat u CLIDE okruženju. Za razliku od prethodne faze u ovoj se kao ciljna platforma koristi procesor CS49834. Struktorno se ovi projekti ne razlikuju, odnosno sadrže iste datoteke. Za pravilno funkcionisanje modula u kom je implementiran algoritam za obradu na ciljnoj platformi, potrebno je obezbediti izvršavanje još nekih modula. Pre svega potreban je operativni sistem, čija je uloga ukratko opisana u jednom od prethodnih poglavlja. Pored OS, u ovom radu koristili su se moduli čijom upotrebotom se obezbeđuje sinhronizacija prilikom snimanja izlaznih vektora na ciljnoj platformi. Bez

korišćenja ovih modula izlazna datoteka bi sadržala određen broj odbiraka tišine ispred odbiraka korisnog signala. Synchronizacijom se obezbeđuje da snimanje izlazne sekvene počne nakon dobavljanja i obrade prvog bloka ulaznih odbiraka.

4.4 Iskorišćenost resursa ciljne platforme

U ovom delu rada izneti su detalji o iskorišćenosti memorije datog DSP procesora, kao i utrošenog procesorskog vremena potrebni za obradu bloka podataka modulom Bass Extract. Iskorišćenost memorije se izražava kao broj zauzetih reči, odnosno broj zauzetih 32-bitnih memorijskih lokacija. Iskorišćenost procesorskog vremena se meri u MIPS-ima, odnosno milionima instrukcija u sekundi. Računanje potrošnje procesorskog vremena se vrši pomoću sledeće formule:

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

Parametar Fs predstavlja frekvenciju odabiranja, broj odbiraka u jednoj sekundi ulaznog signala, koja u ovom radu iznosi 48kHz, a BLOCK_SIZE predstavlja veličinu bloka obrade, u ovom slučaju 16 odbiraka. Pošto simulatorski režim omogućuje kontrolisano izvršavanje programa, moguće je utvrditi tačan broj ciklusa koji je potreban za izvršenje niza instrukcija, odnosno funkcije. Na taj način je izračunat broj ciklusa potreban za izvršenje *Block* rutine u kojoj se vrši obrada jednog bloka učitanih odbiraka. Uvrštavanjem vrednosti u prethodnu formulu dobija se podatak potrošnji procesorskog vremena, odnosno MIPS-a. Podaci o potrošnji memorije, odnosno broju zauzetih lokacija, mogu se pročitati iz datoteke sa ekstenzijom .MAP koja se dobija prilikom generisanja izvršne datoteke.

U sledećoj tabeli (Tabela 4.1) prikazana je potrošnja resursa korišćene platforme, memorije za podatke, programske memorije i maksimalna potrošnja vremena procesora.

X memorija	Y memorija	P memorija	MIPS
31	836	258	8,745

Tabela 4.1 Prikaz potrošnje resursa ciljne platforme

5. Ispitivanje

Ispitivanje i verifikacija modula vršeni su u nakon završetka svake od prethodno opisanih faza. Svrha ovog procesa je dokazivanje da se izlazi generisani modulom slažu sa referentnim izlazima koji se dobijaju uz projekat ili se generišu pomoću referentne aplikacije. Postoji više načina ispitivanja rezultata:

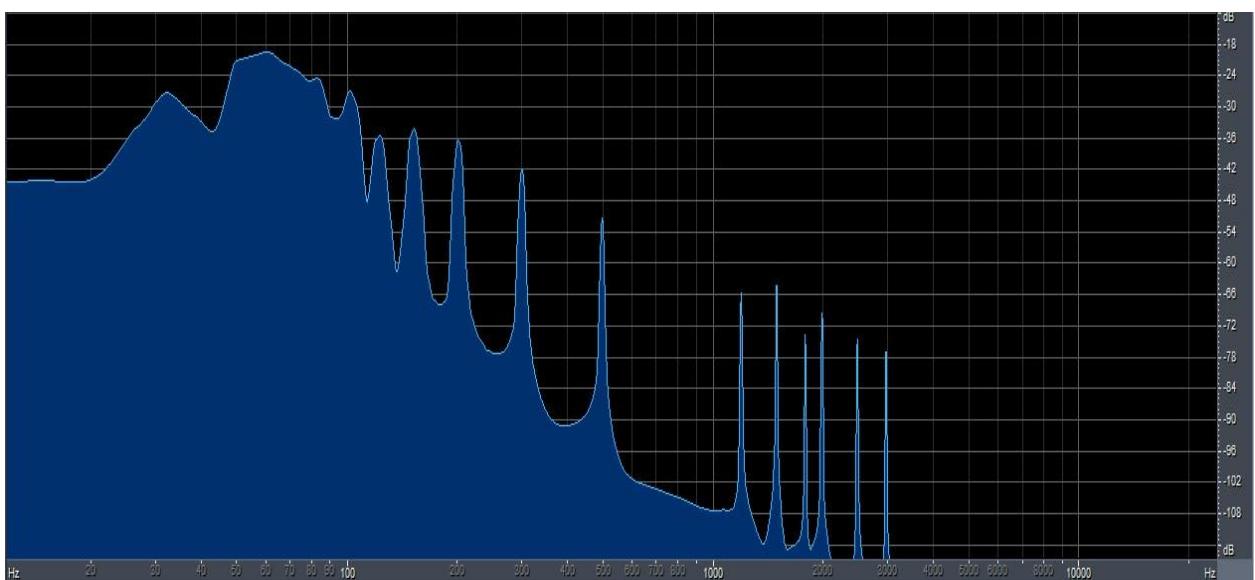
- Ručni bit-identični testovi
- Automatski bit-identični testovi
- Slušni testovi – preslušavanje

Pošto ne postoje zvanični testni vektori, pomoću kojih bi se mogla ispitati ispravnost algoritma, algoritma, pomoću alata Adobe Audition, generisana su četiri osmokanalna test vektora u wav formatu. Prvi test vektor, white_noise_no_lfe.wav, u prvih sedam kanala nosi beli šum, dok je osmi (LFE) kanal prazan. Drugi vektor, white_noise.wav, u svim kanalima nosi beli šum, osim u LFE kanalu u kom je multitonski signal sa niskim frekvencijama do 150 Hz. Drugu grupu testnih vektora čine vektori sa multitonskim signalima u svakom kanalu. Kao i kod prethodne grupe, jedan od vektora, (multi_tone.wav), u svom LFE kanalu nosi niskofrekventni multitonski signal, dok je kod drugog, (multi_tone_no_lfe.wav), LFE kanal prazan. U sledećoj tabeli (Tabela 5.1) dat je pregled frekvencijskih komponenti u pojedinim kanalima. Razlog zašto su izabrane upravo ove frekvencije je taj da se pokaže efikasnost implementiranih filtera. Niskofrekventnim filtriranjem, visokofrekventne komponente (1200, 1500, 1800, 2000, 2500, 3000 Hz) bivaju potisnute, dok se niskofrekventne komponente (50, 60, 70, 80 Hz) potiskuju prilikom visokofrekventnog filtriranja. Ako se posmatra slučaj kada je granična frekvencija postavljena na 80 Hz, prikazano na sledećoj slici (

Slika 5.1), komponente bliske toj frekvenciji neće biti u velikoj meri potisnute prilikom niskofrekventnog filtriranja, razlog za to je što implementirani filtri nemaju idealne karakteristike.

Naziv kanala	Frekvencije [Hz]				
L (Left)	50		1500		
C (Center)	60		2000		
R (Right)	70		2500		
Ls (Left Surround)	80		3000		
Rs (Right Surround)	100		500		
Lb (Left Back)	200		1200		
Rb (Right Back)	300		1800		
LFE (Low Frequency Effect)	30	60	90	120	150

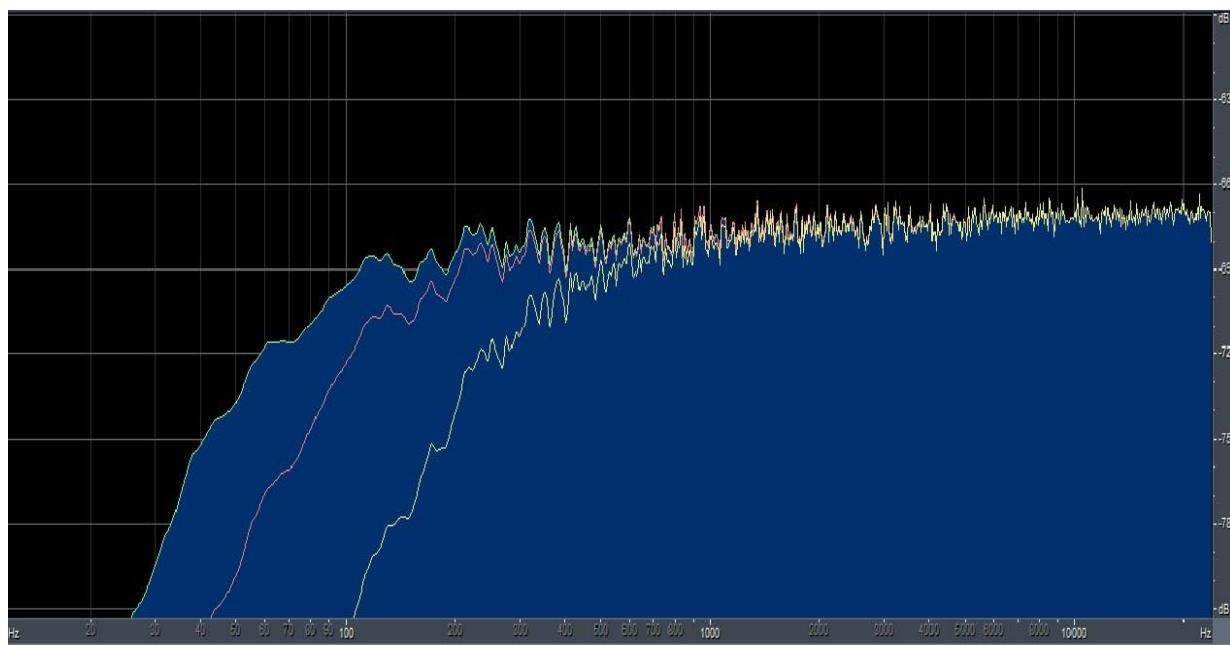
Tabela 5.1 Sadržaj testnog vektora multi_tone.wav



Slika 5.1 Spektralna karakteristika LFE kanala multiton skog testnog vektora

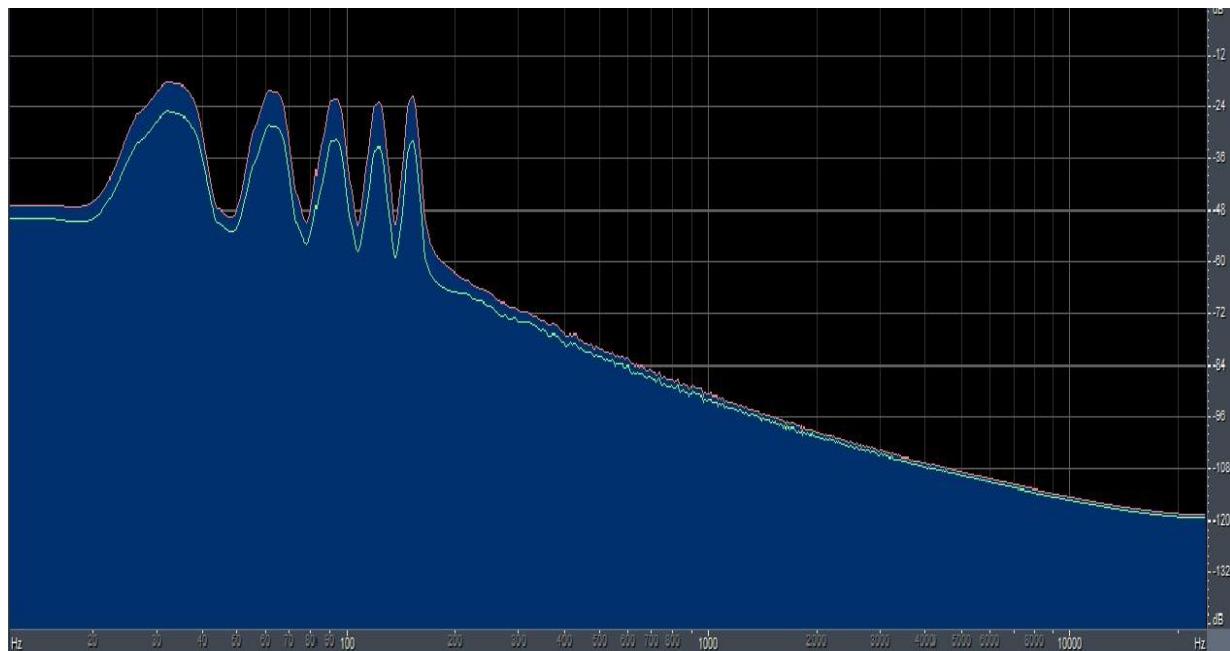
Ovi vektori koriste se kao ulaz za svaku od faza. Na sledećim slikama dati su primeri spektralnih karakteristika izlaznih vektora referentnog projekta generisani korišćenjem različitih vrednosti konfiguracionih parametara.

Na slici (Slika 5.2), prikazane su spektralne karakteristike dobijene visokofrekventnim filtriranjem sadržaja jednog kanala testnog vektora white_noise.wav, za različite granične frekvencije (45 Hz, 80 Hz i 200 Hz). Sa slike se vidi da su niskofrekventne komponente potisnute. Naravno, pošto filtri nemaju idealnu karakteristiku, komponente u blizini graničnih frekvencija nisu potpuno potisnute, odnosno anulirane.



Slika 5.2 Spektralna karakteristika visokofrekventnog filtra za različite vrednosti parametra
Cutoff

Sledeća slika, (Slika 5.3), prikazuje spektralnu karakteristiku LFE kanala izlaznog vektora white_noise.wav, kada je parametar bass_boost uključen (crveno), odnosno isključen (zeleno). Uključivanjem ovog parametra, sadržaj LFE kanala se pre filtriranja dodatno uvećava za 10dB.



Slika 5.3 Spektralna karakteristika za različite vrednosti parametra Bass Boost

Na sledećim slikama (Slika 5.4) prikazan uticaj parametra phase_shift na proces obrade. Na slikama su date karakteristika LFE kanala izlaznog testnog vektora u vremenskom domenu, na desnoj strani slike parametar phase_shift bio je uključen, a na drugoj isključen.



Slika 5.4 Prikaz signala u vremenskom domenu za različite vrednosti parametra Phase Shift

5.1 Ručni bit-identični testovi

Ovaj vid testiranja podrazumeva poređenje izlaza generisanih referentnim programom, sa izlazima generisanim u ostalim fazama na nivou bita. Za potrebe ovakvog testiranja korišćeni su alat PCMCompare.exe i Total Commander. Kao što je i očekivano, izlazi referentnog programa se razlikuju od izlaza generisanih u drugoj i trećoj fazi. Osnovni razlog postojanja ovih razlika jeste prelazak sa aritmetike u pokretnom zarezu na aritmetiku sa nepokretnim zarezom. Primeri ovog vida testiranja dati su na sledećim slikama.

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

D:\test>PCMCompare.exe -w D:\test\white_noise_ref_80_0_0.wav D:\test\whiteNoise_80_0_0_out_sim.wav
Recorded file is 2517368 samples shorter than the reference file!
Test error! Recorded file is 2517368 samples shorter than the reference file!
Max difference is 1 <1 bits, -144.49dB>
Max difference is 1 <1 bits, -144.49dB>
170624 samples compared

Dif<bits> : Samples : PERCENT : First dif
-----+
      1 :     9380 :   5.50% : 0x00000191
-----+
    Error :     9380 :   5.50% : 

D:\test>
```

Slika 5.5 Ručno poređenje referentnih i izlaza simulatorskog projekta

```
D:\>PCMCompare.exe -w D:\test\white_noise_ref_80_0_0.wav D:\test\white_noise_80_0_0_hw.wav
Max difference is 2 <2 bits, -138.47dB
Max difference is 2 <2 bits, -138.47dB
2687992 samples compared
-----  

Dif<bits> | Samples | PERCENT | First dif  

1 | 143808 | 5.35% | 0x00000038  

2 | 26 | 0.00% | 0x00000041  

-----  

Error | 143834 | 5.35%  

D:\>
```

Slika 5.6 Ručno poređenje referentnih i izlaza sa ciljne platforme

```
D:\>PCMCompare.exe -w D:\test\white_noise_80_0_0_hw.wav D:\test\whiteNoise_80_0_0_out_sim.wav
Recorded file is 2517376 samples shorter than the reference file!
Test error! Recorded file is 2517376 samples shorter than the reference file!
Max difference is 2 <2 bits, -138.47dB
Max difference is 2 <2 bits, -138.47dB
170624 samples compared
-----  

Dif<bits> | Samples | PERCENT | First dif  

1 | 316 | 0.19% | 0x00000038  

2 | 16 | 0.01% | 0x00000041  

-----  

Error | 332 | 0.19%  

D:\>
```

Slika 5.7 Ručno poređenje izlaza sa ciljne platforme i izlaza simulatorskog projekta

Na prvoj slici prikazano je poređenje referentnih izlaza sa izlazima iz druge faze, odnosno simulatorskog projekta. Na drugoj slici prikazani rezultati poređenja izlaza generisanih u drugoj i trećoj fazi. Ono što je zanimljivo je to da se u pomenutim primerima porede sekvence različitih dužina. Generisanje cele sekvenca izlaza u drugoj fazi, simulatorskom režimu, traje jako dugo. Da bi se izbeglo čekanje, generiše se samo deo izlaznog vektora. Poređenjem ovih skraćenih sekvenci i referentnih izlaza može se potvrditi ispravnost algoritma razvijenog u drugoj fazi.

5.2 Automatski bit-identični testovi

Prethodno opisan način testiranja je spor, neefikasan i unosi mogućnost ljudske greške. Iz tih razloga često se radi automatizacija testova upotrebom nekog od skript jezika. Za potrebe ovog rada napisane su *batch* skripte. Pisanje ovih skripti u početku oduzima dosta vremena,

međutim kada se napišu postupak testiranja je automatizovan. Time se vreme testiranja znatno skraćuje u odnosu na vreme koje je potrebno za ručno testiranje.

U okviru ovog rada napisane su skripte pomoću kojih se automatizuje proces poređenja referentnih izlaza i izlaza generisanih pomoću ciljne platforme. Ceo postupak može se podeliti u dve celine. Prva celina obuhvata konfigurisanje ciljne platforme i snimanje izlaznih vektora. Konfigurisanje ciljne platforme podrazumeva učitavanje izvršnih (.uld) datoteka svih modula koji se koriste, slanje vrednosti konfiguracionih parametara svakom od modula. Snimanje izlaznih vektora vrši se pomoću alata asio_record.exe. U okviru druge celine izvršava se poređenje referentnih izlaza sa izlazima generisanim na ciljnoj platformi. Poređenje se vrši pomoću alata PCMCompare.exe, a rezultati poređenja upisuju se u tekstualnu datoteku, kako bi se kasnije mogli analizirati.

Na sledećoj slici (Slika 5.8) dat je deo datoteke u kojoj su prikazani rezultati poređenja. Na samom početku naveden je redni broj testa, a zatim nazivi vektora koji se upoređuju. Sledeće informacije prikazuju ukupan broj odbiraka koji su upoređeni, kao i maksimalnu bitsku razliku koja se javlja. U nastavku, dati su detaljni podaci o broju odbiraka kod kojih se javljaju razlike, kao i kolike su te rezlike u bitima. Ovi detalji izraženi su i u procentima.

U ovom primeru, upoređivani su vektori multi_tone_80_0_0.wav (referentni) i multi_tone_80_0_0_hw_out (generisan na ciljnoj platformi). Pri generisanju ovih vektora modul bass_extract je konfigurisan tako da je granična frekvencija 80 Hz, a parametri bass_boost i phase_shift isključeni. Broj upoređenih odbiraka je 2 687 976, a maksimalna bitska razlika je tri bita. Najveći broj odbiraka (1 257 054 odbiraka, 46,77 %) razlikuje se u jednom bitu. Kod 1,83% od ukupnog broja odbiraka primećene su razlike od dva bita. Od ukupnog broja odbiraka, samo 9 (0,00%) se razlikuje u tri bita.

```

Running Test ID: 1
Referent output: c:\Users\andrićn\Documents\Streams\TEST\referent_output_streams\multi_tone_ref_80_0_0.wav
Hw output: d:\Bachelor\Load\bass_extract\test_streams\output\multi_tone_80_0_0_hw_out.wav
-----
Max difference is 4 (3 bits, -132.45dB)
2687976 samples compared
-----
Dif(bits) | Samples | PERCENT | First dif
-----
| 1 | 1257054 | 46.77% | 0x00000041
| 2 | 49190 | 1.83% | 0x000007a6
| 3 | 9 | 0.00% | 0x00001ae1
-----
| Error | 1306253 | 48.60%
-----
#####

```

Slika 5.8 Deo tekstualne datoteke u kojoj su prikazani rezultati testiranja

U prethodno pomenutoj datoteci dat je prikaz rezultata za 18 testnih slučajeva, za različite ulazne vektore i različite konfiguracione parametre modula bass_extract. Za grupu testnih vektora koji nose multitonske signale pojavljuju se razlike maksimalno do tri bita. Međutim, broj odbiraka koji se razlikuju u tri bita je reda 0,00%, što je vrlo mali broj odbiraka. Za drugu grupu testnih vektora, vektora koji nose beli šum u svojim kanalima, primećuju se još manje razlike od maksimalno dva bita.

5.3 Slušni testovi i analiziranje spektralnih karakteristika

Tokom razvoja svake od faza rađeni su slušni testovi. Ova vrsta ispitivanja je naročito značajna u ranim fazama razvoja, jer se na ovaj način može proveriti ispravnost realizovanog algoritma, odnosno da li algoritam obrađuje ulazni tok podataka na očekivani način.

Pored slušnih testova, dobar način za proveru ispravnosti algoritma je i analiza spektralnih karakteristika. Ovaj vid provere pruža mogućnost brzog uočavanja greške, jer se na osnovu spektralne karakteristike izlaznog signala može utvrditi koji deo obrade se ne izvršava kako bi trebalo.

6. Zaključak

U okviru ovog rada implementiran je DSP modul za upravljanje niskofrekventnim sadržajem na procesoru CS49834 firme Cirrus Logic.

Sama implementacija izvršena je u nekoliko faza. Prva faza podrazumevala je analiziranje referentnog koda i razumevanje implementiranog algoritma. Sledeći korak se odnosio na prilagođenje referentnog programa, odnosno uklapanje celina iz kojih se on sastoji. Radno okruženje u ovoj fazi bio je Microsoft Visual Studio 2010.

Druga faza podrazumeva promenu programskog jezika i radnog okruženja. U ovoj fazi koristi se asemblerски jezik ciljne platforme, a razvojno okruženje postaje CLIDE. Najbitnija karakteristika prelaska sa prve na drugu fazu je prelazak na aritmetiku sa nepokretnim zarezom. Ova faza se može podeliti u dva dela. Cilj prvog dela bio je postizanje iste funkcionalnosti koja je realizovana u referentnom projektu. Provera ispravnosti vršena je slušnim testovima i analizom spektralnih karakteristika. Kada je ovaj cilj postignut, prešlo se na drugi deo koji podrazumeva optimizaciju implementiranog rešenja. Kod je prilagođen tako da se što je moguće više iskoriste mogućnosti platforme za paralelno izvršavanje više operacija u okviru jedne instrukcijske reči. Treća faza podrazumeva prelazak iz simulatorskog režima na izvršavanje na ciljnoj platformi. Da bi se omogućio pravilan rad implementiranog modula, u spremi sa njim se izvršavaju još neki moduli.,

Na kraju je izvršeno ispitivanje i verifikacija implementiranog modula proverom bit-identičnosti između izlaza generisanih u pojedinim fazama. Napisane su skripte pomoću kojih se automatski izvršava provera bitske razlike između izlaza generisanih referentnim programom, i izlaza generisanih na ciljnoj platformi. Uočene su maksimalne razlike od tri bita, što je za module završne obrade (eng. *Post-Processing Module*) zadovoljavajuće. Međutim, najveći broj odbiraka se razlikuje u jednom bitu, što se može pripisati promeni aritmetike.

Dalja unapređenja ovog modula odnosila bi se na još veći stepen optimizacije, u cilju bržeg izvršavanja i trošenja manje memorije. Ipak, ova realizacija se uklapa u ograničenja ciljne platforme i ispunjava zahteve za izvršenje u realnom vremenu tako da dodatne optimizacije nisu neophodne.

7. Literatura

- [1] V. Kovačević, M. Popović, M. Temerinac, N. Teslić, „*Arhitekture i algoritmi digitalnih signal procesora 1*“, FTN izdavaštvo, Novi Sad, 2005
- [2] M. Temerinac, S. Berber, Ž. Lukač, „*Osnovi algoritama i struktura DSP I*“, Univerzitet Novi Sad Fakultet Tehničkih Nauka, Novi Sad, 2013
- [3] „*Arhitekture i algoritmi DSP II – praktikum za laboratorijske vežbe*“, Novi Sad 2013
- [4] A. Grimani, „*Bass Management and the LFE Channel*“ , Sound & Vision, August,2005
- [5] „*What is LFE channel?*“, Dolby Laboratories, Inc 2000
- [6] „*Frequently asked questions about Dolby Digital*“ , Dolby Laboratories, Inc 2000
- [7] „*5.1-Channel Music Production Guidelines*“, Dolby Laboratories, Inc 2003
- [8] „*CS498xx_Datasheet*“, Cirrus Logic, Inc., March, 2013