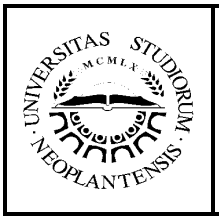


УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

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: **Милена Милошевић**
Број индекса: **Е12647**

Тема рада: **Реализација HTTP Live Streaming протокола за Андроид базирани дигитални ТВ пријемник**

Ментор рада: **Проф. Др. Никола Теслић**

Нови Сад, 2012

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

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:		
Идентификациони број, **ИБР**:		
Тип документације, **ТД**:		Монографска документација
Тип записа, **ТЗ**:		Текстуални штампани материјал
Врста рада, **ВР**:		Завршни (Bachelor) рад
Аутор, **АУ**:		Милена Милошевић
Ментор, **МН**:		Проф. Др. Никола Теслић
Наслов рада, **НР**:		Реализација HTTP Live Streaming протокола за Андроид базирани дигитални ТВ пријемник
Језик публикације, **ЈП**:		Српски / латиница
Језик извода, **ЈИ**:		Српски
Земља публиковања, **ЗП**:		Република Србија
Уже географско подручје, **УГП**:		Војводина
Година, **ГО**:		2012
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)		7/27/0/0/15/0/0
Научна област, **НО**:		Електротехника и рачунарство
Научна дисциплина, **НД**:		Рачунарска техника
Предметна одредница/Кључне речи, **ПО**:		Телевизија, HTTP Live Streaming протокол, Андроид ОС
УДК		
Чува се, **ЧУ**:		У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, **ВН**:		
Извод, **ИЗ**:		У овом раду је приказано једно рјешење реализације HTTP Live Streaming протокола (HLS) за Андроид базирани дигитални ТВ пријемник. Рјешење је реализовано на Marvell BG2 SoC платформи.
Датум прихватања теме, **ДП**:		
Датум одбране, **ДО**:		
Чланови комисије, **КО**:	Председник:	
	Члан:	Потпис ментора
	Члан, ментор:	

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

KEY WORDS DOCUMENTATION

Accession number, **ANO**:	
Identification number, **INO**:	
Document type, **DT**:	Monographic publication
Type of record, **TR**:	Textual printed material
Contents code, **CC**:	Bachelor Thesis
Author, **AU**:	Milena Milošević
Mentor, **MN**:	PhD Nikola Teslić
Title, **TI**:	One implementation of HTTP Live Streaming protocol for Android based DTV platform
Language of text, **LT**:	Serbian
Language of abstract, **LA**:	Serbian
Country of publication, **CP**:	Republic of Serbia
Locality of publication, **LP**:	Vojvodina
Publication year, **PY**:	2012
Publisher, **PB**:	Author's reprint
Publication place, **PP**:	Novi Sad, Dositeja Obradovica sq. 6
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)	7/27/0/0/15/0/0
Scientific field, **SF**:	Electrical Engineering
Scientific discipline, **SD**:	Computer Engineering, Engineering of Computer Based Systems
Subject/Key words, **S**/**KW**:	Television, HTTP Live Streaming, Android OS
UC	
Holding data, **HD**:	The Library of Faculty of Technical Sciences, Novi Sad, Serbia
Note, **N**:	
Abstract, **AB**:	One implementation of HTTP Live Straming protocol on Android DTV plarform is described in this paper. Solution is implemented on Marvell BG2 SoC platform.
Accepted by the Scientific Board on, **ASB**:	
Defended on, **DE**:	
Defended Board, **DB**: President:	
Member:	Menthor's sign
Member, Mentor:	

SADRŽAJ

SPISAK SLIKA

SKRAĆENICE

ABS – *Adaptive Bitrate Switching*, Adaptivna promjena bitske brzine

API – *Application Programming Interface,* Programska sprega

CDN – *Content Delivery Network,* Računarska mreža za isporuku sadržaja

DTV – *Digital Television,* Digitlna televizija

H.264 – *MPEG4 Part 10 (=AVC- Advanced Video Coding)*

HbbTV – *Hybrid Broadcast Broadband Television,* Hibridna emitersko-širokopojasna televizija

MPEG2 TS – *MPEG2 Transport Stream*, MPEG2 digitalni tok podataka

MPEG DASH – *MPEG Dynamic Adaptive Streaming over HTTP,* Adaptaciono emitovanje digitalnog toka podataka preko HTTP-a(eng. HyperText Transfer Protocol)

MSF – *Media Segment File,* Kratka datoteka koja sadrži tok podataka (sliku i zvuk)

NAT – *Network Address Transaltion,* Preslikavanje mrežne adrese

OS – *Operating System,* Operativni sistem

PID – *Packet ID,* Identifikacioni broj paketa

RTP – *Real-Time Transport Protocol,* Protokol za isporuku slike i zvuka preko IP mreže

RTSP – *Real-Time Streaming Protocol,* Protokol za kontrolu isporuke slike i zvuka

SoC – *System on Chip*, Sistem u integrisanom kolu

URL – *Uniform Resource Locator,* Web adresa

VOD – *Video On Demand,* Video na zahtjev

1. Uvod

U ovom radu je prikazano jedno rješenje realizacije HTTP Live streaming protokola (HLS) za Android bazirani digitalni TV prijemnik. Rješenje je realizovano na Marvell Berlin Generation 2 System on Chip (Marvell BG2 SoC) platformi.

Opisana je tipična arhitektura za HTTP Live Streaming protokol. Protokol je realizovan korišćenjem i mijenjanjem postojećih modula na takav način da se dalja funkcionalnost može lako realizovati. Pod daljom funkcionalnošću se podrazumijeva uvođenje adaptacione logike kako bi adaptivna promjena bitske brzine-ABS (eng. adaptive bitrate switching) bila podržana. Takođe je izvršeno i ispitivanje pomoću posebne Java aplikacije kako bi se utvrdila ispravnost realizacije.

Ovaj rad je sačinjen od 7 poglavlja.

U prvom poglavlju je opisan sadržaj rada, a sedmo sadrži spisak korišćene literature.

Drugo poglavlje opisuje šta je to što je dovelo do razvoja digitalne televizije – DTV (eng. digital television), zašto je HTTP baziran tok podataka (eng. stream) postao popularan i tipičnu arhitekturu HTTP Live Streaming-a.

U trećem poglavlju dat je opis funkcionalnosti HTTP Live Streaming-a koji treba da se realizuje kao i specifičnosti vezane za platformu koje su uticale na samu realizaciju ovog protokola. Takođe, dat je i koncept rješenja.

Četvrto poglavlje sadrži detaljan opis same realizacije HLS-a, a peto poglavlje opisuje način ispitivanja i verifikovanja te realizacije.

Šesto poglavlje sarži kratak opis šta je urađeno u ovom radu i koji su pravci daljeg razvoja.

2. Teorijske osnove

U ovom poglavlju opisani su razvoj digitalne televizije, uzroci popularnosti HTTP baziranog toka podataka i tipična HTTP Live Streaming arhitektura.

2.1 Razvoj digitalne televizije i Interneta

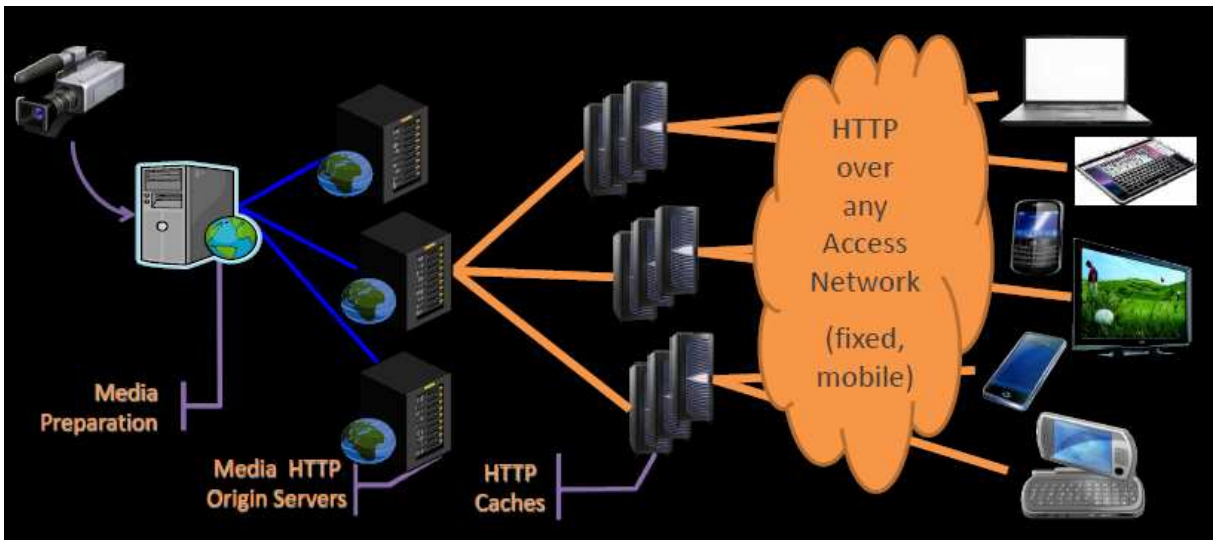
Razvoj televizije tokom prošlog vijeka blisko prati naučni napredak u tehnologiji. Pojava velikih ekrana, visoke rezolucije i niske cijene Plasma i LCD televizijskih uređaja i prijemnika obezbijedili su veliki korak u razvoju televizije - prelazak sa analogne na digitalnu televiziju. DTV(digitalna televizija) nudi mnoga poboljšanja u odnosu na analognu. Najvažnije poboljšanje je prenos slike i zvuka sa dodatnim inforamcijama u digitalnom obliku. Emitovani TV kanal može prenositi više TV programa i veliki broj drugih podataka u isto vrijeme: prevodi, teletekst, višestruki audio tokovi i drugi digitalni podaci.

DTV je ubrzala razvoj običnog TV prijemnika. TV prijemnik se promijenio od običnog CRT (eng. Cathode Ray Tube) ekrana sa skupom namjenskih analognih i digitalnih komponenti do ugradjenog SoC-a sa tankim kućištem i LCD ili Plasma monitorom. Sa modernim višejezgarnim CPU-ovima (eng. Central processing unit) i taktom preko 1GHz ovi „mali računari“ (arhitektura moderne potrošačke elektronike liči na personalne računare) su i više nego sposobni da izvršavaju dodatne složene aplikacije paralelno sa prikazom DTV sadržaja.

Postojali su mnogi pokušaji da se iskoriste mogućnosti savremenog DTV uređaja, da bi se obezbijedila i druga funkcionalnost sem gledanja televizije. Većina proizvođača DTV uređaja realizovala je kao rješenja programsku podršku zasnovanu na linux operativnom sistemu. Jedno od takvih rješenja je korišćenje Android operativnog sistema u DTV prijemnicima. Ova ideja je prihvatljiva iz više razloga: Android operativni sistem je

dizajniran, realizovan i zasnovan na linux operativnom sistemu za namjenske uređaje; Android operativni sistem nudi mnoštvo aplikacija dostupnih za skidanje i instalaciju. Nedostatak podrške za DTV u Android OS i jedan prijedlog realizacije razmotreni su u [1].

S druge strane, proteklih godina Internet je postao dostupan skoro svima i polako preuzima sve veću ulogu u životu svakodnevnih ljudi. Veliki broj aplikacija koje nudi Android operativni sistem kao i aplikacija koje dolaze iz digitalnog toka podataka (kao što je HbbTV) zahtijevaju pristup Internetu, iz više razloga. Jedan od razloga je emitovanje slike i zvuka preko Interneta, koje zahvaljujući ovakvoj arhitekturi postaje dostupno korisnicima putem TV prijemnika. Prije naglog porasta propusne moći Interneta početkom prve decenije 21. vijeka, reprodukcija slike i zvuka preko Interneta se zanivala na principu „skini pa onda gledaj“ (eng. download and then play)[2]. Rastom propusne moći, kao i razvojem boljih algoritama za kompresiju slike i zvuka, emitovanje slike i zvuka je postalo moguće. Sam termin emitovanje slike i zvuka(eng. streaming) se koristi da opiše prenos podataka preko računarske mreže kao stabilnog neprekidnog toka(eng. stream), dozvoljavajući da reprodukcija slike i zvuka počne dok se naredni podaci i dalje primaju. Ovo je u suprotnosti sa principom „skini pa onda gledaj“, gdje reprodukcija počinje nakon što se svi podaci prime. Trenutni početak reprodukcije je glavna prednost isporuke slike i zvuka emitovanjem, jer korisnik više ne mora da čeka da se skidanje čitavog sadržaja završi, što može trajati i satima u slučaju spore veze.



Slika 2.1 Iskorišćenje postojećih HTTP poslužilaca za isporuku slike i zvuka

Jedan od protokola preko koga se vrši emitovanje slike i zvuka je HTTP [3]. HTTP omogućava krajnjem korisniku da gleda video bez potrebe za preuzimanjem čitavog sadržaja prije njegovog puštanja. Ovakav način dostavljanja slike i zvuka je postao standard na Internetu iz dva razloga. Prvo, solidna Internet veza je danas dostupna bilo gdje, bilo kada i na

bilo kom uređaju. Drugo, korišćenje HTTP protokola ne uzrokuje NAT(eng. Network Address Translation) i firewall upite kao što je to slučaj sa drugim protokolima za prenos slike i zvuka kao što je RTP/RTSP (eng. Real-Time Transport Protocol/Real-Time Streaming Protocol).

Takođe, HTTP bazirana isporuka pruža mogućnost korišćenja standardnih HTTP poslužilaca i standardnih HTTP poslužilaca posrednika(eng. caches) (ili jeftinih poslužilaca uopšte) za isporuku slike i zvuka, tako da slika i zvuk mogu biti isporučeni sa CDN-a(eng. Content Delivery Network) ili bilo koje druge standardne mreže poslužilaca. Ovo podrazumijeva korišćenje postojeće infrastrukture, što smanjuje operativne troškove (Slika 2.1). [4]

2.2 Tipična HTTP Live Streaming arhitekura

Dvije najpoznatije tehnike za emitovanje slike i zvuka preko HTTP protokola su HLS (eng. HTTP Live Streaming) i MPEG DASH. Ovaj rad se bavi realizacijom HTTP Live Streaming protokola na digitalnom TV prijemniku koji je zasnovan na Android operativnom sistemu. HTTP Live Streaming je protokol za prenos ograničenog ili videa na zahtjev-VOD (eng. Video on Demand – prethodno snimljeni sadržaj) i neograničenog (eng.live broadcast – emitovanje uživo) toka podataka.

HTTP Live Streaming se sastoji od 3 dijela: dijela na strani poslužioca, dijela za distribuciju i klijentske programske podrške [5] :

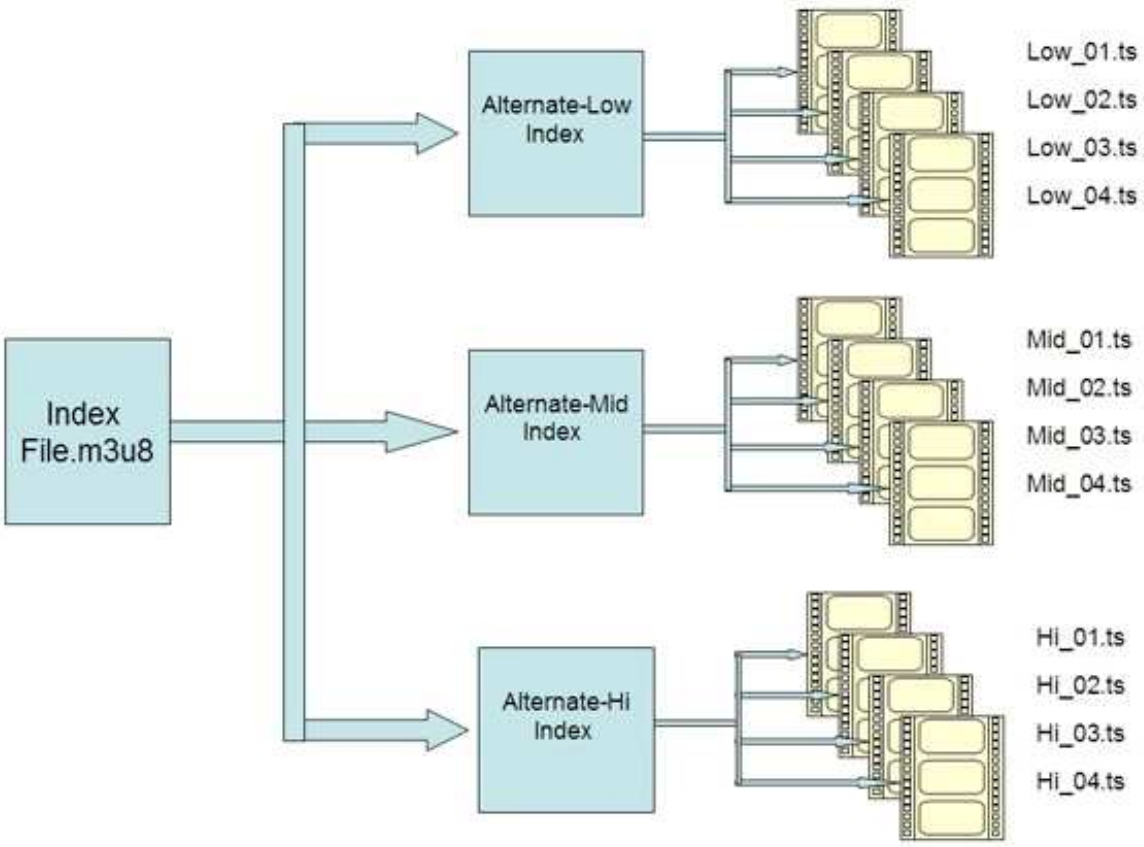
- Dio na strani poslužioca je zadužen za preuzimanje ulaznog sadržaja, njegovo digitalno kodiranje, pakovanje u oblik pogodan za isporuku i pripremanje tog sadržaja za isporuku.
- Dio za distribuciju (isporuku) se sastoji od standardnih web poslužioca. Oni su zaduženi za prihvatanje klijentskih zahtjeva i isporuku prethodno pripremljenog toka podataka i pratećih resursa klijentu.
- Klijentska programska podrška je odgovorna za slanje zahtjeva za odgovarajućim sadržajem, preuzimanjem tog sadržaja i njegovim ujedinjavanjem tako da dobijeni sadržaj može biti predstavljen korisniku na odgovarajući način.

Obično, enkoder fizičke arhitekture uzima sadržaj sa video-audio ulaza, koduje ga u H.264 video i AAC audio i izbacuje u MPEG2 TS formatu, koji se zatim dijeli u serije kratkih datoteka koje sadrže sliku i zvuk – MSF-ove (eng. Media segment files) programskim djeliocem toka podataka (eng. software segmenter). Te datoteke se stavljaju na web

poslužioce. Djelilac toka podataka takođe stvara i održava indeksu datoteku koja sadrži listu MSF-ova. URL indeksne datoteke se objavljuje na web poslužiocu. Klijentska programska podrška čita indeksnu datoteku, zatim zahtijeva MSF-ove redom kako su izlistani u indeksnoj datoteci i prikazuje ih bez pauza ili praznina između tih dijelova.

HTTP Live Streaming šalje sliku i zvuk u serijama MSF-ova, čije trajanje je obično oko 10 sekundi. Indeksna datoteka ili lista za reprodukciju (eng. playlist) daje klijentu URL-ove do MSF-ova. Lista za reprodukciju se može periodično osvježavati kako bi se vršila reprodukcija emitovanja uživo, gdje se novi MSF-ovi stalno proizvode i dodaju u listu [6].

Indeksne datoteke mogu referencirati alternativne tokove podataka. Reference se koriste da podrže isporuku istog sadržaja različitog kvaliteta za različite propusne opsege ili uređaje. Klijentska programska podrška koristi posebnu logiku da odredi pravo vrijeme za prelazak na drugi alternativni tok podataka. Trenutno, ta logika se zasniva na najnovijim trendovima u mjerenju mrežnog protoka. Na slici ispod je prikazan primjer indeksne datoteke koji ukazuje na alternativne tokove podataka uključujući specijalno označenu listu drugih indeksnih datoteka.



Slika 2.2 Struktura indeksnih datoteka i MSF-ova kada postoje alternativni tokovi podataka

Specifikacija HTTP Live Streaming protokola je IETF Internet-Draft [6].

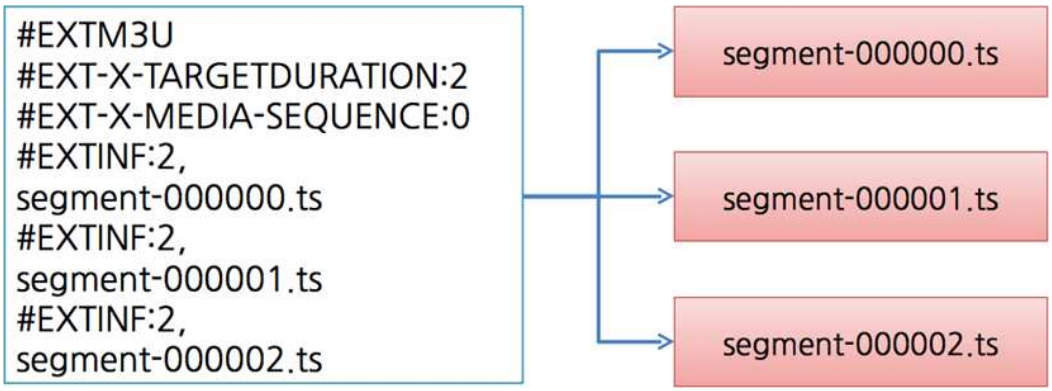
3. Analiza zahtjeva i koncept rješenja

Zahtjevi koje je potrebno ispuniti prilikom realizacije su zahtjevi HTTP Live Streaming protokola kao i zahtjevi vezani za ciljnu platformu i operativni sistem.

3.1 Zahtjevi HTTP Live Streaming protokola

Potrebno je obezbijediti rukovanje m3u8 datotekama kao i skidanje MSF-ova jer se rad HLS-a bazira na sledećem:

Indeksne datoteke su u m3u8 formatu (prošreni m3u format). Svaka m3u8 datoteka počinje posebnom direktivom #EXTM3U što govori programu za reprodukciju da su dolazni podaci u m3u8 formatu. Postoje i druge direktive koje se mogu naći u [6]. U m3u8 datotekama redom su nabrojani MSF-ovi ili druge indeksne datoteke sa pridruženim im propusnim opsegom. Primjer je prikazan na slici ispod:

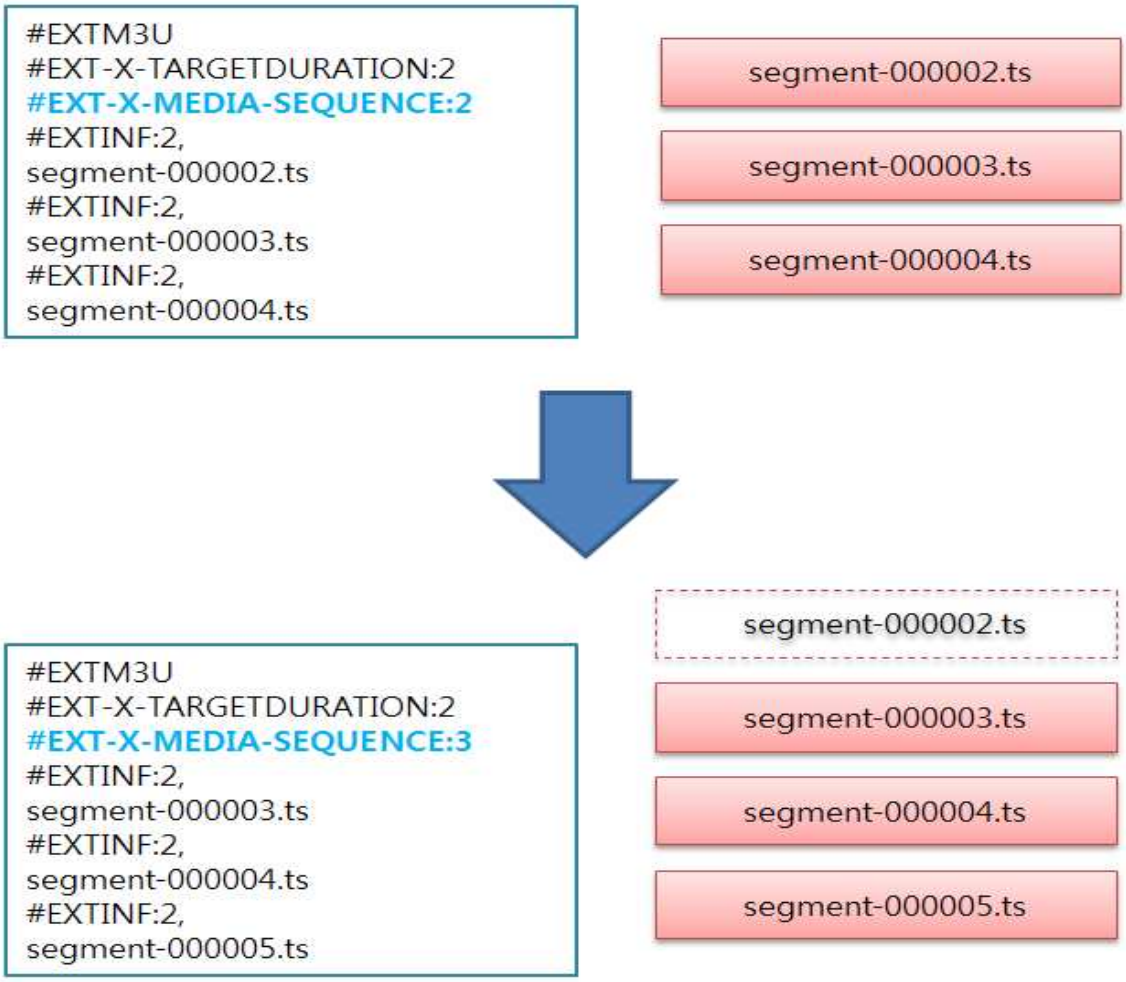


Slika 3.1 Primjer m3u8 datoteke

HTTP Live Streaming protokol podržava dva tipa sesija: emitovanje uživo i emitovanje na zahtjev.

Kod emitovanja na zahtjev svi MSF-ovi su nabrojani u indeksnoj datoteci i indeksna datoteka se učitava samo na početku.

Za sesije uživo kako se novi MSF-ovi stvaraju i postaju dostupni tako se i indeksna datoteka ažurira. Ažurirana indeksna datoteka sadrži nove MSF-ove, stariji MSF-ovi se obično uklone. Ažurirana indeksna datoteka predstavlja klizajući prozor (eng. Sliding window) kroz kontinualni tok podataka. Kada je novi dio za reprodukciju spreman (MSF), indeksna datoteka se ažurira tako što se dodaje putanja do novog MSF-a, a uklanja putanja do najstarijeg. Na ovaj način indeksna datoteka uvijek sadrži x poslednjih MSF-ova. Preporučuje se da x bude veće od 3 tako da klijent može pauzirati, ponovo pustiti(eng. resume) i premotati(eng. rewind) barem u periodu trajanja poslednja 3 MSF-a (podrazumijevana vrijednost je 10 sekundi po MSF-u). [7].



Slika 3.2 Ažuriranje m3u8 datoteke kod emitovanja uživo

Sledeće funkcionalnosti HTTP Live Streaming-a treba da budu podžane

- reprodukcija slike i zvuka
- dva tipa sesija : emitovanje uživo i emitovanje na zahtjev.

- pauziranje (eng. pause), nastavak reprodukcije(eng. resume), skakanje na zadatu poziciju u datoteci (eng. seeking) kod VOD-a, a kod emitovanja uživo pauziranje, nastavljanje reprodukcije i premotavanje(eng. rewind) barem u periodu trajanja poslednja 3 MSF-a (podrazumijevano 30 sekundi).
- način realizacije treba da bude takav da omogućava laku naknadnu realizaciju adaptacione logike (eng. adaptive bitrate switching)

3.2 Specifičnosti vezane za ciljnu platformu

Osnovna ideja pri realizaciji klijentske programske podrške za HTTP Live Streaming je da bude tako realizovana, u okviru već postojeće realizacije digitalnog TV prijemnika baziranog na Android OS , da omogućuje lako proširenje, tj. dodavanje novih mogućnosti, funkcionalnosti.

Kod realizovanja klijentske programske podrške ideja je da se ne uvodi nova sprega prema krajnjem korisniku već se zadržava postojeća, što omogućava korisnicima laku upotrebu na način na koji su već navikli[8]. Sve što je potrebno da bi puštali video preko HTTP Live Streaming-a je da kao putanju proslijede URL koji se odnosi na HTTP Live Streaming tamo gdje su do sada proslijeđivali putanju do drugog video sadržaja. Dalji način rukovanja je identičan kao i kod reprodukcije slike i zvuka bilo kog tipa.

Android kao OS od verzije 3.0 pa naviše podržava veći dio funkcionalnosti HTTP Live Streaming-a[9]. Obzirom da se koristi Android koji je prilagođen za ciljnu platformu, javlja se problem kod korišćenja ugrađenog Androidovog programa za reprodukciju slike i zvuka(eng. player) za HTTP Live Streaming, jer se kod njega dekodovanje vrši u okviru programske podrške. Iz tog razloga u ovoj realizaciji DTV platforme bazirane na Android OS, ugrađeni program za reprodukciju slike i zvuka se ne koristi, već se koristi program za reprodukciju koji je specifičan za samog proizvođača i podržava dekodovanje toka podataka u okviru same fizičke arhitekture što doprinosi na brzini.

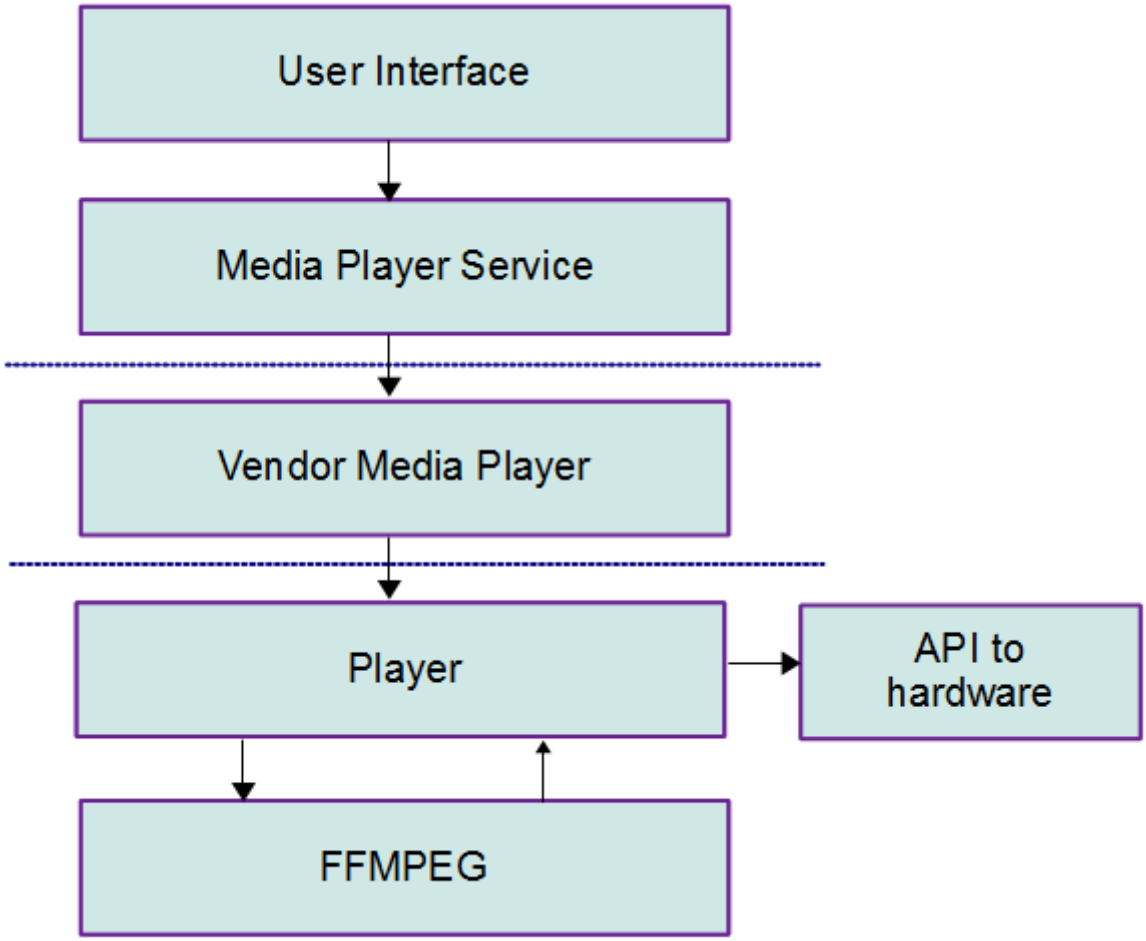
3.3 Koncept rješenja

Slojevi programske podrške koji se koriste za realizaciju HTTP Live Streaming-a od kojih će neki biti mijenjani prikazani su na slici 3.3.

User Inteface predstavlja spregu prema korisniku o kojoj je već bilo riječi i ona nije mijenjana.

Media Player Service je postojeći Androidov service koji prihvata komande od korisnika i odlučuje koji program za reprodukciju slike i zvuka će instancirati kako bi korisnik

bio uslužen. U našem slučaju to je Vendor Media Player, program za reprodukciju koji je specifičan za samog proizvođača.



Slika 3.3 Arhitektura programske podrške za realizaciju HLS-a

Vendor Media Player na osnovu proslijeđenog URL-a stvara jedan od svojih programa za reprodukciju koji zna da rukuje datim sadržajem. U slučaju reprodukcije HLS-a to je FFplayer.

FFPlayer koristi određene funkcije FFMPEG biblioteke koja je takođe prilagođena za datu ciljnu platformu. Pomoću FFMPEG biblioteke dobavlja sve potrebne informacije o toku podataka kao i sliku i zvuk, koje zatim prosleđuje u dekoder fizičke arhitekture preko odgovarajućeg API-ja (eng. Application Programming Interface) ciljne platforme [10].

4. Programsko rješenje

Da bi se realizovao HLS prilagođeni su postojeći programi za reprodukciju i koristi se postojeća biblioteka - FFMPEG.

4.1 Sprega ka korisniku

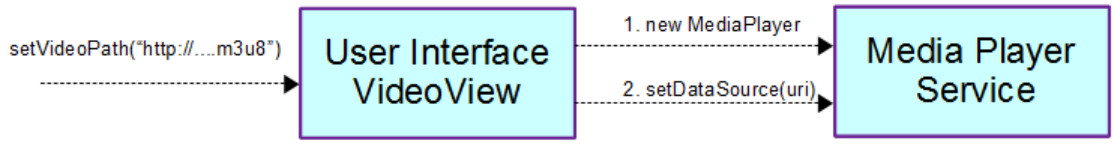
Programska sprega ka korisniku je ostala ista, kao i u dosadašnjoj realizaciji reprodukcije slike i zvuka [8]. Za HLS su podržane osnovne funkcionalnosti:

- reprodukcija slike i zvuka (eng. play)
- zaustavljanje reprodukcije (eng. stop)
- pauziranje reprodukcije (eng. pause)
- nastavak reprodukcije (eng. resume)
- skakanje na određenu poziciju u datoteci, reprodukcija sa zadate pozicije (eng. seek)

Poslednje 3 su u potpunosti podržane kod VOD-a, a kod emitovanja uživo se može pauzirati, nastaviti reprodukciju i premotati (eng. rewind) barem u periodu trajanja poslednja 3 MSF-a.

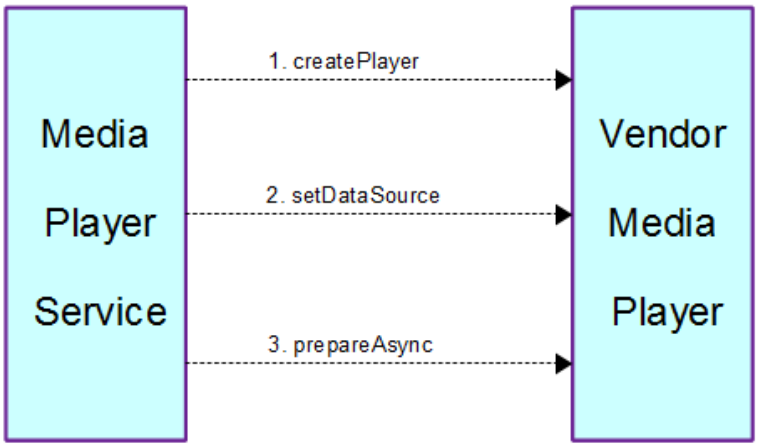
4.2 Reprodukcija slike i zvuka

Da bi korisnik pustio video, potrebno je da navede putanju do videa, tj. apsolutni URL(ili URI), koji se odnosi na HTTP Live Streaming i to putanju do indeksne datoteke. Pri tome se stvara Media Player Service, koji je Androidov ugrađeni service za programe za reprodukciju slike i zvuka i poziva se njegova funkcija setDataSource kojom se prosljeđuje URI.



Slika 4.1 Komunikacija između programske sprege ka korisniku i servisa

Media Player Service na osnovu proslijeđenog URI-ja stvara odgovarajući program za reprodukciju koji zna da rukuje sadržajem na koji ukazuje URL. U ovom slučaju ne stvara se neki od androidovih programa za reprodukciju slike i zvuka, nego se stvara program za reprodukciju koji je specifičan za ciljnu platformu (Vendor Media Player). Zadatak Vendor Media Player-a je da prepozna URL koji se odnosi na HTTP Live streaming i da stvori odgovarajući program za reprodukciju koji zna da pusti taj sadržaj. Takođe njegov zadatak je da grafičke slojeve učini providnim(eng. transparent), kako bi se na ekranu jasno prikazao sadržaj koji se reprodukuje u video ravni.



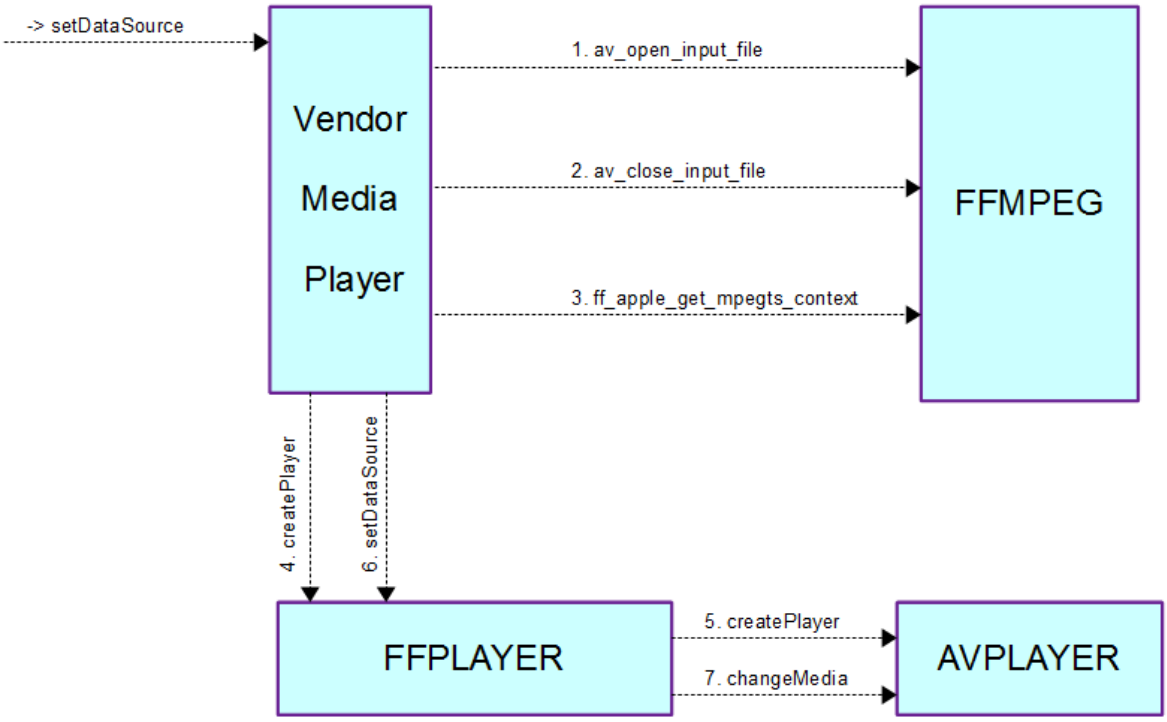
Slika 4.2 Komunikacija između servisa i modula Vendor Media Player

Pri samom stvaranju Vendor Media Player-a izvršavaju se neka početna podešavanja bliska samoj fizičkoj arhitekturi. Zatim se pozivaju sledeće funkcije VendorMediaPlayer-a:

- setDataSource – vrši dodatna podešavanja sve do fizičke arhitekture i podešavanja istog kako bi se reprodukovao novi tok podataka,
- prepareAsync – pripremi sve za reprodukciju i reprodukuje sadržaj .

4.2.1 Pripreme za prelazak na reprodukciju novog toka podataka

Poziv funkcije setDataSource vrši dodatna podešavanja sve do fizičke arhitekture i podešavanja istog kako bi se sve pripremilo za prelazak na reprodukciju trenutnog toka podataka(slika 4.3).



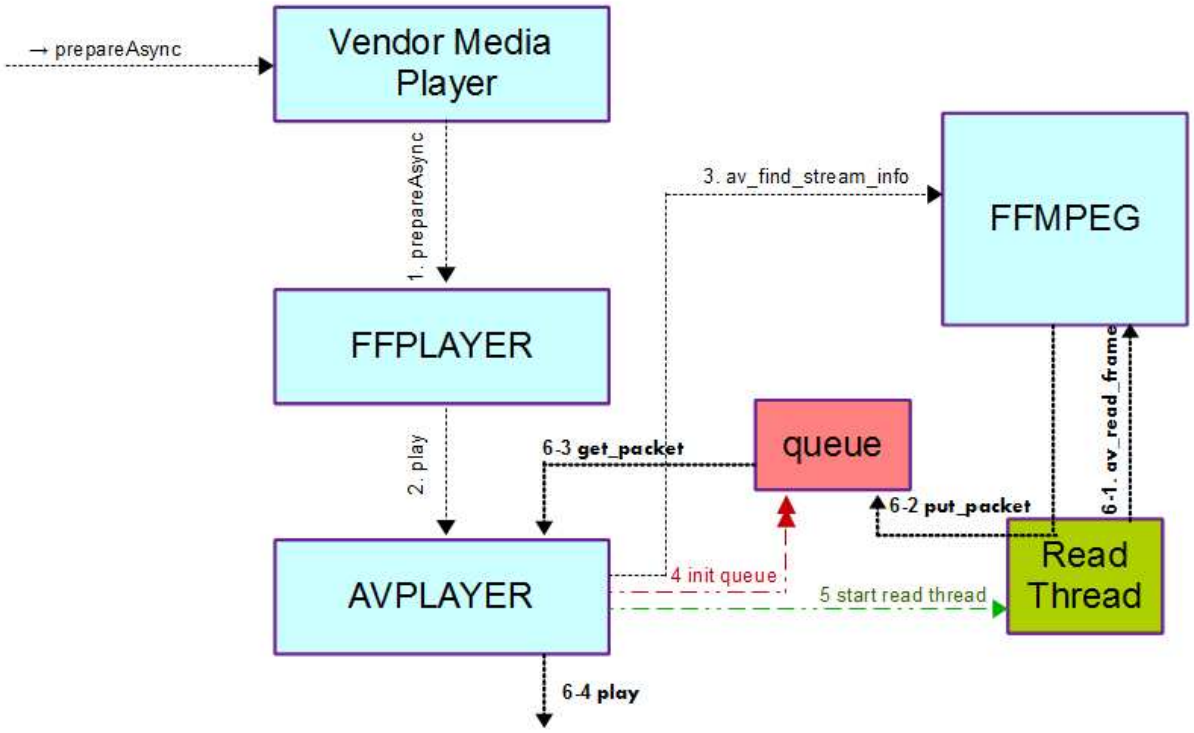
Slika 4.3 Komunikacija između modula nakon poziva funkcije setDataSource

Vendor Media Player poziva funkcije koje nudi FFMPEG biblioteka kako bi dobio više informacija o toku podataka koji treba da reprodukuje i kako bi dobio sam tok podataka. Poziva se funkcija av_open_input_file u kojoj se vrši ispitivanje toka podataka. Skidaju se m3u8 datoteke, vrši se njihovo parsiranje i smještanje informacija u strukture na dva nivoa. Prvi nivo odgovara indeksnoj datoteci koja pokazuje na druge indeksne datoteke. Drugi nivo odgovara indeksnoj datoteci koja odgovara jednom propusnom opsegu(slika 2.2). U slučaju da nema alternativnih tokova podataka, ovakva struktura se zadržava(zadržava se jedan nivo koji je nepotreban). Nivoi sadrže relevantne podatke za podržane propusne opsege. Takođe, sadrže pokazivače ka MSF-ovima. Nakon poziva ove funkcije Vendor Media Player dobija informacije o slici i zvuku u okviru strukture AVFormatContext. VendorMediaPlayer sada zna da je riječ o HTTP Live Streming-u. Kako bi znao koji format slike i zvuka dalje da proslijedi kako bi se izvršila dodatna podešavanja vezana za niže slojeve programske podrške i samu ciljnu platformu, of FFMPEG biblioteke traži dodatne podatke za sliku i zvuk (tj. za mpegts). Zatim stvara odgovarajući program za reprodukciju koji zna da reprodukuje taj

sadržaj. U ovom slučaju to je FFPlayer, koji stvara program za reprodukciju AVplayer koji pored ostalog preko određenih API-ija zna da rukuje fizičkom arhitekturom(inicijalizacija, puštanje sadržaja, razna podešavanja za promjenu sadržaja i sl.). Zatim se poziva setDataSource sa prikupljenim informacijama koji poziva funkciju changeMedia AVPlayer-a, kako bi se izvršile sve pripreme da bi novi sadržaj mogao da se reprodukuje.

4.2.2 Podešavanja za reprodukciju i reprodukcija

Poziv prepareAsync funkcije inicijalizuje i priprema sve potrebno za reprodukciju slike i zvuka i započinje reprodukciju, i zatim asinhronim događajom javlja da se sadržaj počeo reprodukovati(slika 4.4).



Slika 4.4 Komunikacija između modula nakon poziva funkcije prepareAsync

Kada se pozove funkcija play() modula AVplayer, on poziva av_find_stream_info kako bi dobio informacije o toku podataka koji treba da reprodukuje (format slike i zvuka, audio i video PID-ove(eng. packet ID) i sl.).Vrši dodatna podešavanja(kao što su postavljanje početnog PTS-a (eng. Presentation Time Stamp) i sl.). Nakon toga inicijalizuje se bafer iz koga će AVPlayer da čita pakete i da ih dalje šalje na reprodukciju. Takođe, inicijalizuje se i startuje nit koja će da puni taj isti bafer paketima pozivajući funkciju FFMPEG-a av_read_frame. FFMPEG interno zna koji paket da vrati. Vodi evidenciju o tome koji je na redu, šta treba skidati preko HTTP GET zahtjeva[3], i sledeći paket za reprodukcija vraća niti za čitanje, koja puni bafer. AVplayer će čitati iz bafera kada dobije notifikaciju da je bafer

popunjen do određene veličine. Čita jedan po jedan paket i šalje dalje kroz postojeći API ka fizičkoj arhitekturi[10] kako bi se vršila njegova reprodukcija. Koraci 6-1;6-2 i 6-3;6-4 se izvršavaju paralelno sve vrijeme dok traje reprodukcija.

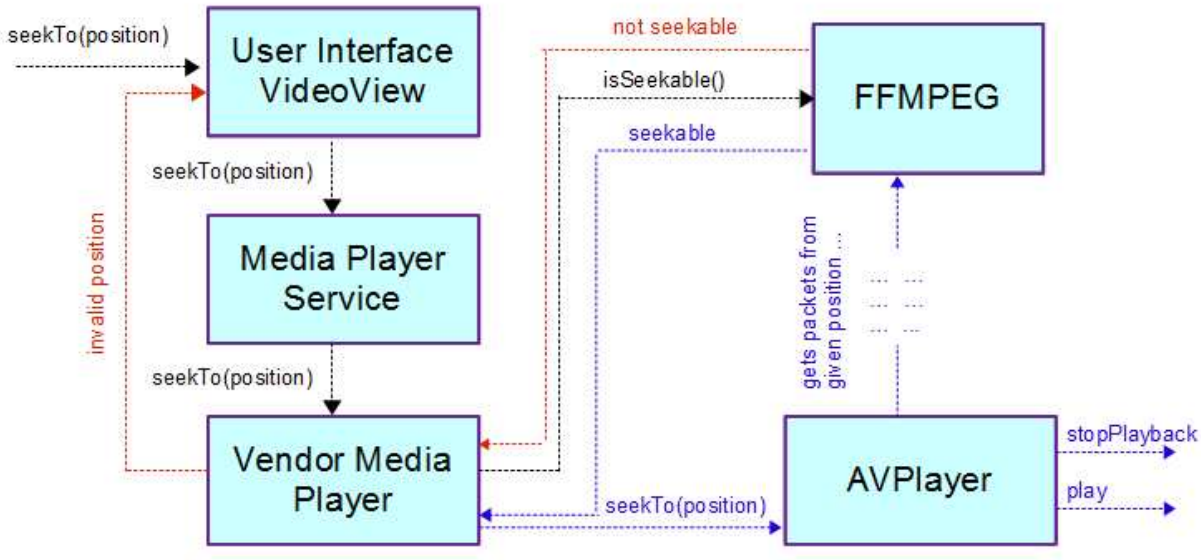
4.3 Zaustavljanje reprodukcije slike i zvuka

Pri zaustavljanju reprodukcije korisnik poziva funkciju stopPlayback, što prolazi kroz sve slojeve programske podrške do AV player-a koji vrši zaustavljanje niti za čitanje, kao i deinicijalizaciju buffer-a, i samo zaustavljanje reprodukcije na ciljnoj platformi.

4.4 Skakanje na određenu poziciju u datoteci tokom reprodukcije(eng. seeking)

Skakanje na određenu poziciju je u potpunosti podržano kod videa na zahtjev, a kod emitovanja uživo najmanje u periodu trajanja poslednja tri MSF-a.

U Vendor Media Player-u pri pozivu funkcije seek vrši se računanje apsolutne i/ili relativne pozicije na koju treba da se skoči. Pri tome se provjerava da li trenutni fajl čija reprodukcija se vrši podržava to skakanje. Poziva se funkcija isSeekable(). Ako je u pitanju video na zahtjev i ako pozicija na koju se treba izvršiti skok ne izlazi iz okvira trajanja fajla, skakanje je podržano, i poziva se seek funkcija AVPlayer-a koji prvo zaustavi reprodukciju slike i zvuka,a zatim ponovo pokrene reprodukciju od zadate pozicije u fajlu. Pri samom pokretanju reprodukcije FFMPEG zna da dostavlja digitalni tok podataka od zadate pozicije.



Slika 4.5 Komunikacija između modula tokom skakanja na određenu poziciju u datoteci

Nakon pozivanja funkcije isSeekable ako je u pitanju emitovanje uživo skakanje će biti podržano ako tražena pozicija ne izlazi iz perioda trajanja poslednja tri MSF-a, tj onoliko MSF-ova koliko je veličina klizajućeg prozora[7].

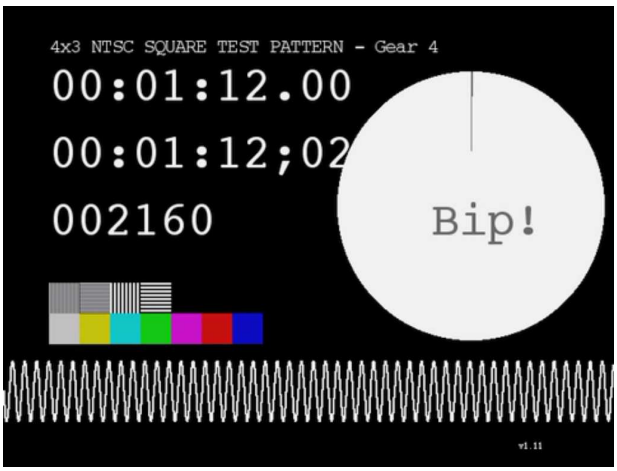
4.5 Pauziranje i nastavak reprodukcije slike i zvuka

Kao i skakanje i pauziranje je u potpunosti podržano kod videa na zahtjev, dok se kod emitovanja uživo može pauzirati najduže u vremenu trajanja poslednja 3 MSF-a.

Pozivi ovih funkcija se proslijeđuju AVPlayer-u koji zna da rukuje njima u skladu sa podacima koje dobija iz FFMPEG-a.

5. Ispitivanje i verifikacija

Za potrebe ispitivanja napravljena je aplikacija u Java programskom jeziku, koja koristi funkcije postojeće sprege ka korisniku i tako simulira razne situacije koje bi se mogle desiti u realnom sistemu. Svaka provjera se sastoji od poziva niza funkcija, nakon čega se dobijeni rezultat upoređuje sa očekivanim. Sve provjere rađene su pomoću postojećeg Apple-ovog toka podataka za ispitivanje HTTP Live Streaming-a [5], koji je prikazan na slici ispod:



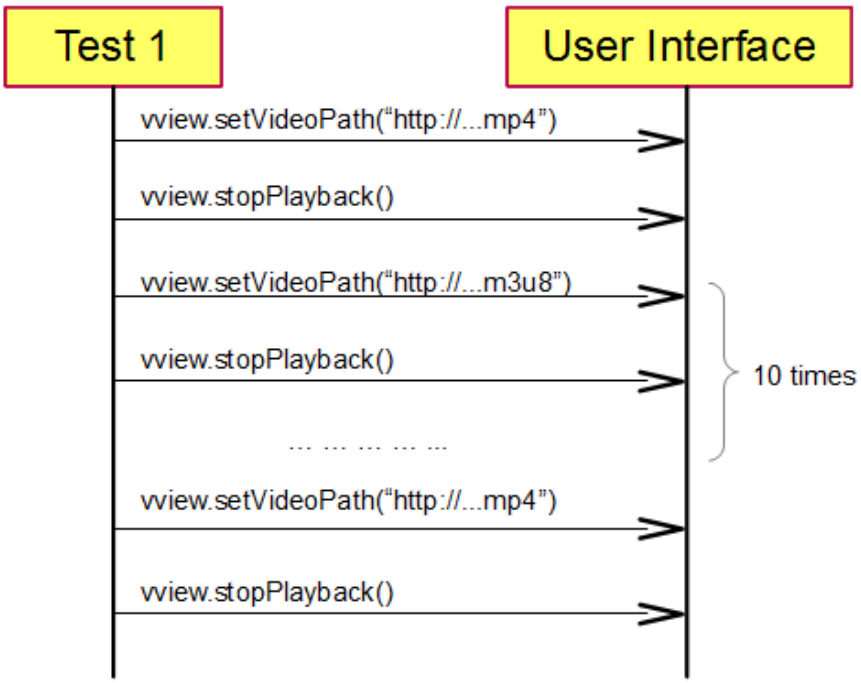
Slika 5.1 Apple-ov tok podataka za ispitivanje HLS-a

5.1 Ispitivanje ispravnosti inicijalizacije i deinicijalizacije

Ova provjera vrši višekratnu inicijalizaciju i deinicijalizaciju. Na slici 5.2 je prikazan redoslijed poziva funkcija.

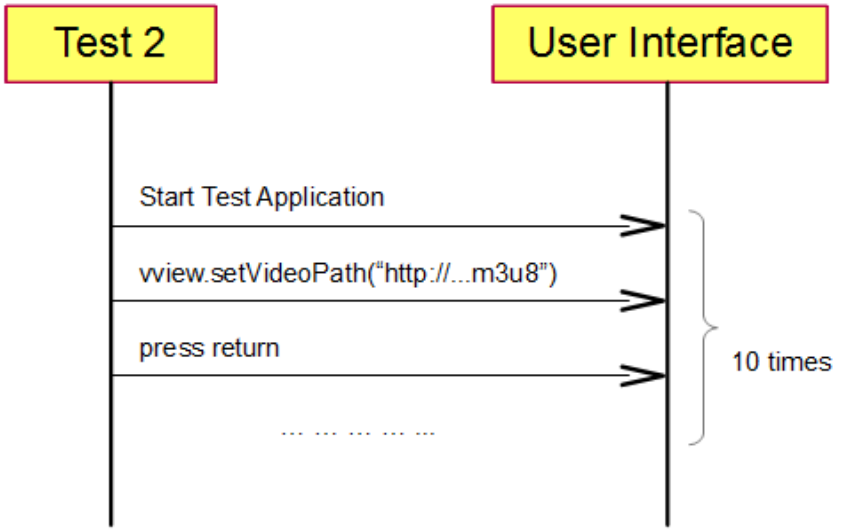
Provjerava se inicijalizacija za reprodukciju HTTP Live Streaming sadržaja nakon reprodukcije nekog drugog sadržaja , zatim višekratna inicijalizacija i deinicijalizacija same HTTP Live Streaming reprodukcije, kao i reprodukcija drugog sadržaja nakon HTTP Live Streaming sadržaja. Ispravnost se potvrđuje vizuelno, tako što se prvo vidi reprodukcija mp4

videa, zatim višestruka reprodukcija i prestanak HTTP Live Streaming videa i na kraju ponovo reprodukcija mp4 videa.



Slika 5.2 Provjera br.1

Također se provjerava ispravnost deinicijalizacije nakon izlaska iz Android aplikacije, bez prethodnog eksplicitnog zaustavljanja reprodukcije i provjera ispravnosti inicijalizacije istog nakon ponovnog ulaska u aplikaciju.

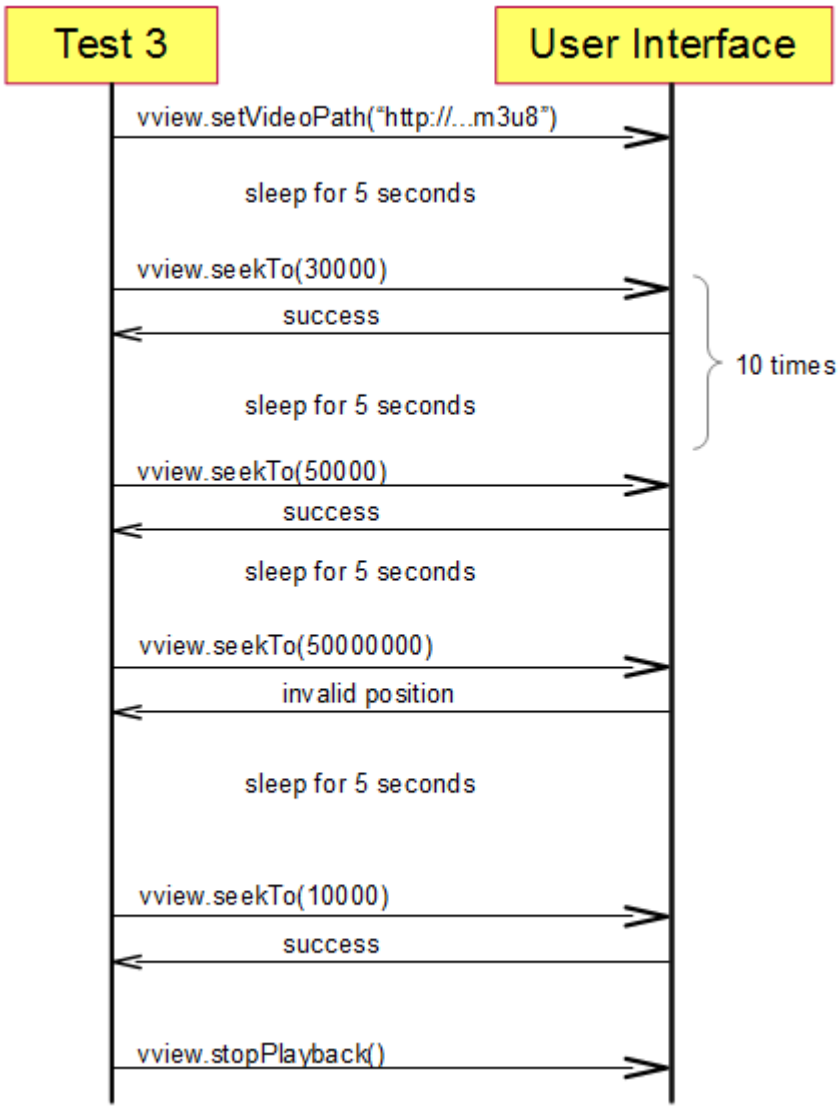


Slika 5.3 Provjera br. 2

Ispravnost ove provjere se također potvrđuje vizuelno. Nakon pritiska na dugme return aplikacija se gasi, a reprodukcija zaustavlja, a nakon ponovnog pokretanja kreće reprodukcija slike i zvuka, kao što je i očekivano.

5.2 Ispitivanje ispravnosti skakanja na zadatu poziciju

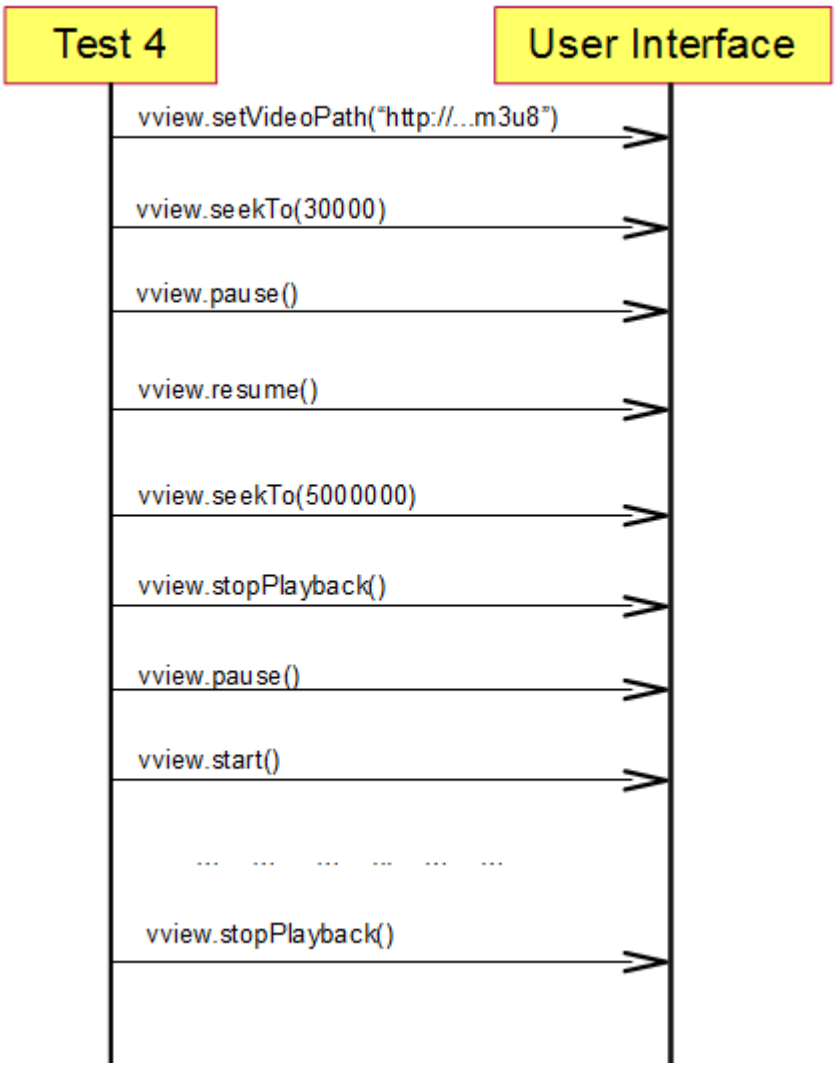
Vrši se provjera ispitivanja ispravnosti skakanja na zadatu pozicuiju. Provjerava se skakanje na poziciju koja nije ispravna, kao i na pozicije koje su ispravne.



Slika 5.4 Provjera br. 3

5.3 Ispitivanje robustnosti HLS-a

Funkcije se pozivaju nasumično bez unaprijed određenog redoslijeda, čime se ispituje otpornost na otkaze i sama robustnost HLS realizacije.



Slika 5.5 Provjera br. 4

Ponašanje realizacije je u skladu sa očekivanim rezultatima.

6. Zaključak

U ovom radu opisano je jedno rješenje realizacije HTTP Live Streaming protokola na Android operativnom sistemu prilagođenom za Marvell Berlin BG2 SoC platformu.

Da bi se postigla reprodukcija toka podataka koji se emituje kao HTTP Live Streaming, prilagođeni su postojeći moduli za reprodukciju toka podataka (MediaPlayerService, Vendor MediaPlayer, FFPlayer, AVPlayer).

Rješenje je ispitivano na ciljnoj fizičkoj arhitekturi. Ispitana je funkcionalnost kao i robustnost realizacije, a svi rezultati su uspješni.

Ovo rješenje je moguće sa lakoćom modifikovati kako bi se ubacila podrška za ABS (eng. adaptive bitrate switching), u cilju što bolje reprodukcije slike i zvuka krajnjem korisniku [11]. Da bi se to postiglo neophodno je dodati adaptacionu logiku u FFMPEG biblioteku ili dodati adaptacionu logiku u sam program za reprodukciju.

7. Literatura

[1] M.Vidakovic, N.Teslic, T.Maruna, and V.Mihic: *Android4TV: a proposition for integration of DTV in Android devices,* IEEE 30th International Conference on Consumer Electronics (ICCE), Las Vegas, January 2012, pp. 441-442

[2] James F. Kurose, Keith W.Ross: *Computer Networking: A Top Down Approach*, 5th edition. Addison Wesley, Chapter 7, 2009

[3] R.Fielding, UC Irvine, J.Gettys, J.Mogul, H.Frystyk, L.Masinter, P.Leach and T.Berners-Lee: *Hypertext Transfer Protocol – HTTP-1.1*, IETF Network Working Group, Request for Comments: 2616, April 1999

[4] Tomas Stockhammer*: Dynamic Adaptive Streaming over HTTP – Design Principles and Standards*

[5] Apple Inc.: *HTTP Live Streaming Overview*, 2009, Online PDF: https://developer.apple.com/library/ios/#documentation/NetworkingInternet/Conceptual/StreamingMediaGuide

[6] R.Pantos, *HTTP Live Streaming: draft-pantos-http-live-streaming-07*, 2009, Online PDF: http://tools.ietf.org/pdf/draft-pantos-http-live-streaming-07.pdf

[7] Andrew Fecheyr-Lippens: *A review of HTTP Live Streaming*

[8] Android Developers, *VideoView Interface*, http://developer.android.com/reference/android/widget/VideoView.html

[9] Android Developers, *Android 3.0 Platform Highlights*, http://developer.android.com/sdk/android-3.0-highlights.html

[10] Marvell, *88DE3010 Software API User Manual*

[11] K. Lazic, M. Milosevic, M. Kovacev, N. Smiljkovic, *"One Implementation of adaptive streaming over HTTP on Android DTV platform*", IEEE 2nd International Conference on Consumer Electronics (ICCE), Berlin, September 2012, accepted