ЗБОРНИК РАДОВА
ФАКУЛТЕТА ТЕХНИЧКИХ НАУКА

Едиција: Техничке науке - зборници
Година: XXIII
Број: 1/2008

УНИВЕРЗИТЕТ У НОВОМ САДУ

ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

ЗБОРНИК РАДОВА
ФАКУЛТЕТА ТЕХНИЧКИХ НАУКА

Нови Сад, 2009.

Едиција: “ТЕХНИЧКЕ НАУКЕ - ЗБОРНИЦИ”

Година: XXIII *Свеска:* 1

Издавач: Факултет техничких наука Нови Сад

Главни и одговорни уредник: Проф. др Илија Ћосић, декан Факултета техничких наука у Новом Саду

Штампа: ФТН - Графички центар ГРИД, Трг Доситеја Обрадовића 6, Нови Сад

Техничка обрада: Графички центар ГРИД

Штампање одобрио: Савет за издавачко-уређивачку делатност ФТН у Новом Саду

Председник Савета за издавачко уређивачку делатност:
Др Радомир Фолић, проф. емеритус

CIP-Каталогизација у публикацији
Библиотека Матице српске, Нови Сад

378.9(497.113)(082)
62

ЗБОРНИК радова Факултета техничких наука / главни и одговорни уредник Илија Ћосић. - Год. 7, бр. 9 (1974)-1988/1989, бр. 19/20 ; Год. 23, бр. 1 (2008)-. - Нови Сад : Факултет техничких наука, 1974-1989;2008-. - Илустр. ; 30 цм. - (Едиција: Техничке науке - зборници)

Двомесечно

ISSN 0350-428X

COBISS.SR-ID 58627591

ПРЕДГОВОР

Поштовани читаоци,

Пред вама је први број обновљеног часописа „Зборник радова Факултета техничких наука“, који поново излази после 18 година паузе. Одлуком Наставно-научног већа Факултета техничких наука од 26.11.2008. год. ревитализован је часопис, који је покренут давне 1960. год., одмах по оснивању Машинског факултета у Новом Саду, као „Зборник радова Машинског факултета“. Часопис после 12 годишта (бројева), 1974. год. мења назив у „Зборник радова Факултета техничких наука“ пратећи прерастање Машинског факултета у Факултет техничких наука. У том периоду у часопису се објављују научни и стручни радови, резултати истраживања професора, сарадника и студената ФТН-а, али и аутора ван ФТН-а, тако да часопис постаје значајно место презентације најновијих научних резултата и достигнућа. Нарастањем материјалних проблема, те доласком несрећних догађаја на нашим просторима, часопис привремено престаје да излази, тако да је последњи број објављен 1991. год., као двоброј/двогодишњак - Год. 20/21, 1990/1991.

Сређивањем стања у друштву и консолидовањем наше државе, Републике Србије, све квалитетнијом организацијом научног рада, реорганизацијом наставног процеса и увођењем читавог низа нових струка, те широким укључивањем ФТН-а у домаће и међународне научне пројекте и истраживања, јавља се потреба и стичу се услови за оживљавањем „Зборника радова Факултета техничких наука“, као јединственог форума за презентацију научних и стручних достигнућа. Значајне промене у структури високог образовања, везане за имплементацију Болоњске декларације, усвајање нове и активне улоге студената у процесу образовања и њихово све шире укључивање у стручне и истраживачке пројекте, као и покретање нових докторских студија, доносе потребу да и ови, веома значајни и вредни резултати, постану доступни академској јавности.

Из тих разлога се НН веће ФТН-а одлучило да од новембра 2008. год. у облику пилот пројекта, а од фебруара 2009, као сталну активност, уведе презентацију најважнијих резултата свих дипломских-мастер радова студената ФТН-а у виду кратког рада у „Зборнику радова Факултета техничких наука“. Поред студената дипломских-мастер студија, часопис је отворен и за студенте докторских студија, као и за прилоге аутора ван ФТН-а.

Зборник ће излазити у два облика – електронском на web сајту ФТН-а (www.ftn.ns.ac.yu) и папирном, који је пред вама. Електронска верзија часописа ће излазити једном месечно, док папирна 5-6 пута годишње.

У овом првом броју штампани су радови студената, сад већ дипломираних инжењера – мастера, који су дипломирали у периоду 1.10. – 15.11.2008. год., а који се промовишу на Светог Саву, 27. 01. 2009. год. То су углавном прилози студената, који су већ објавили радове из својих дипломских-мастер радова на некој од домаћих научних конференција (ЕТРАН, ТЕЛФОР, ЦИРЕД-СРБИЈА), јер је таква традиција постојала за

студенте електротехнике и рачунарства. Ови радови су овде дати као репринт уз мање визуелне корекције. Поред њих, публикован је и један рад из области Инжењерског менаџмента.

Уредништво се нада да ће и професори и сарадници ФТН-а и других институција наћи интерес да публикују своје резултате истраживања у облику регуларних радова у овом часопису. Ти радови ће бити објављивани на енглеском језику због пуне међународне видљиости и проходности презентираних резултата.

У плану је да часопис својим редовним изласком и високим квалитетом, привуче пажњу и ускоро постане довољно препознатљив и цитиран да може да стане раме-уз-раме са водећим часописима и нађе своје место на СЦИ листи. Тиме ће се и на овај начин остварити мото Факултета техничких наука :

„Високо место у друштву најбољих“

Уредништво

SADRŽAJ

Radovi iz oblasti: Elektrotehnika i računarstvo

strana

Radovi iz oblasti: Inženjerski menadžment

JEDNO REŠENJE SISTEMA ZA AUTOMATSKO ISPITIVANJE TELEVIZIJSKOG SIGNALA NA BAZI PROGRAMABILNIH SEKVENCIJALNIH MREŽA

Aleksandar Tomašević, Fakultet Tehničkih Nauka, Novi Sad
Zoran Krajačević, Miloš Nikolić, MicronasNIT, Novi Sad

Sadržaj – *U ovom radu je prikazano jedno rešenje fizičke arhitekture sistema na bazi sekvencijalnih programabilnih mreža koja uz odgovarajuću programsku podršku služi za automatsko ispitivanje TV prijemnika.*

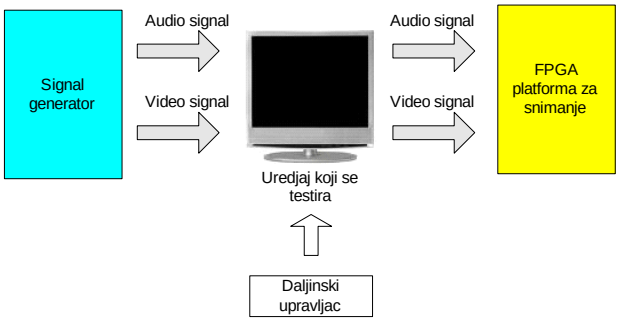
1. UVOD

Neprestanim razvojem novih tehnologija dolazi do pojave sve složenijih multimedijalnih uređaja. Najnovije generacije multimedijalnih uređaja u sebi objedinjuju veliki broj različitih funkcija, počevši od prikaza audio i video zapisa u širokom spektru formata pa preko velikog broja različitih tipova veza (USB –Universal Serial Bus, LAN – Local Area Network itd.) pa sve do objedinjavanja doskora nezavisnih uređaja u jednu celinu (multimedijalni centri itd.). Zbog sve veće kompleksnosti takvih uređaja javlja se potreba za njihovim sveobuhvatnim ispitivanjem. Potreba za ispitivanjem uređaja nikako nije mogla da zaobiđe i jednu od najprofitabilnijih industrija – televizijsku industriju, koja se nalazi u neprestanom razvoju. U današnje vreme, televizija ne
obuhvata samo prenos audio-vizuelnog sadržaja već pruža i širok spektar drugih vrsta usluga čija funkcionalnost mora biti u svakom trenutku pouzdana. Osnovna ideja sastoji se u tome što se TV prijemnik posmatra kao crna kutija sa unapred poznatim ulazima (I/O – Input/Output), mogućnošću kontrolisanja TV prijemnika, kao i sa skupom referentnih audio/video izlaza.

2. METODOLOGIJA ISPITIVANJA

Centralni deo sistema predstavlja uređaj koji se ispituje (slika 1). Pomoću generatora slike/zvuka, na ulaz ispitivanog uređaja se dovode odgovarajući signali slike/zvuka, a korišćenjem komandnih uređaja, ispitivanom uređaju se izdaju komande.

Ispitivani uređaj na izlazu daje sliku, koja se obradjuje uz pomoću FPGA (*Field Programmable Gate Array*-sekvencijalna programabilna mreža) platforme i snima u memoriju. Zahvaljujući modulu implementiranom u FPGA, PC računar memoriju na platformi vidi kao masovnu memoriju, te joj vrlo lako pristupa i izčitava njen sadržaj. Programska podrška za ispitivanje upoređuje dobijenu sliku sa referentnom, i na osnovu toga zaključuje da li se ispitivani uređaj ponaša u skladu sa očekivanjima ili ne.

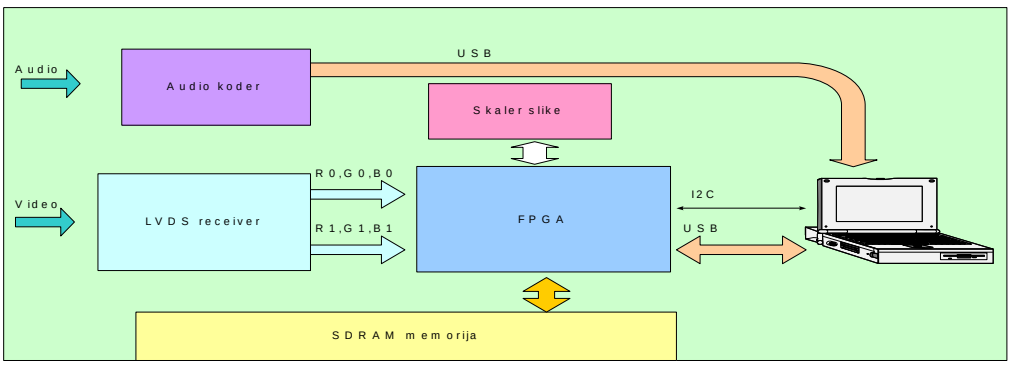


slika 1. *Prikaz metodologije ispitivanja*

Automatsko ispitivanje realizovano na ovaj način predstavlja brz i efikasan način ispitivanja, koji isključuje učešće ljudi tokom izvršavanja ispitivanja. Prednost ispitivanja nekog uređaja kada se on posmatra kao crna kutija je u tome što nije potrebno poznavati njegovu imlpementaciju. Sama programska podrška za ispitivanje zna samo ispravnu reakciju ispitivanog uređaja na odgovarajuću komandu. Osoba koja upravlja ispitivanjem ne mora posedovati detaljno znanje o sistemu koji se ispituje, pošto se ispitivanje za ciljnu platformu izvršavaju sa stanovišta krajnjeg korisnika.

3. FIZIČKA ARHITEKTURA FPGA PLATFORME

Fizička arhitektura sistema je prikazana na slici 2.



slika 2. *Fizička arhitektura sistema*

Napomene:
a) Rad je proistekao iz diplomskog-master rada Aleksandra Tomaševića. Mentor je bio prof.dr Miroslav Popović.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

Centralno mesto sistema zauzima sekvencijalno programabilna mreža Xilinx Spartan3. Spartan3 familija predstavlja naslednika ranije generacije iz Spartan familije FPGA - Spartan-IIE familije sa povećanim brojem logičkih kola, kapacitetom interne RAM memorije, ukupnim brojem ulazno/izlaznih nožica, i ukupnim nivoom performansi kao i poboljšanim funkcijima za kontrolisanje takta.

Za prihvat LVDS (*Low Voltage Differential Signal*- niskovoltni diferencijalni signal) zadužen je LVDS prijemnik koji vrši pretvaranje LVDS toka podataka u 67 bita CMOS/TTL podataka. U dual modu (podaci o dva piksela se istovremeno pojavljuju na izlazu iz prijemnika) maksimalna frekvencija rada je 135 MHz, pri čemu se emituje 67 bita RGB podataka po efektivnoj stopi od 945 Mbps po kanalu.

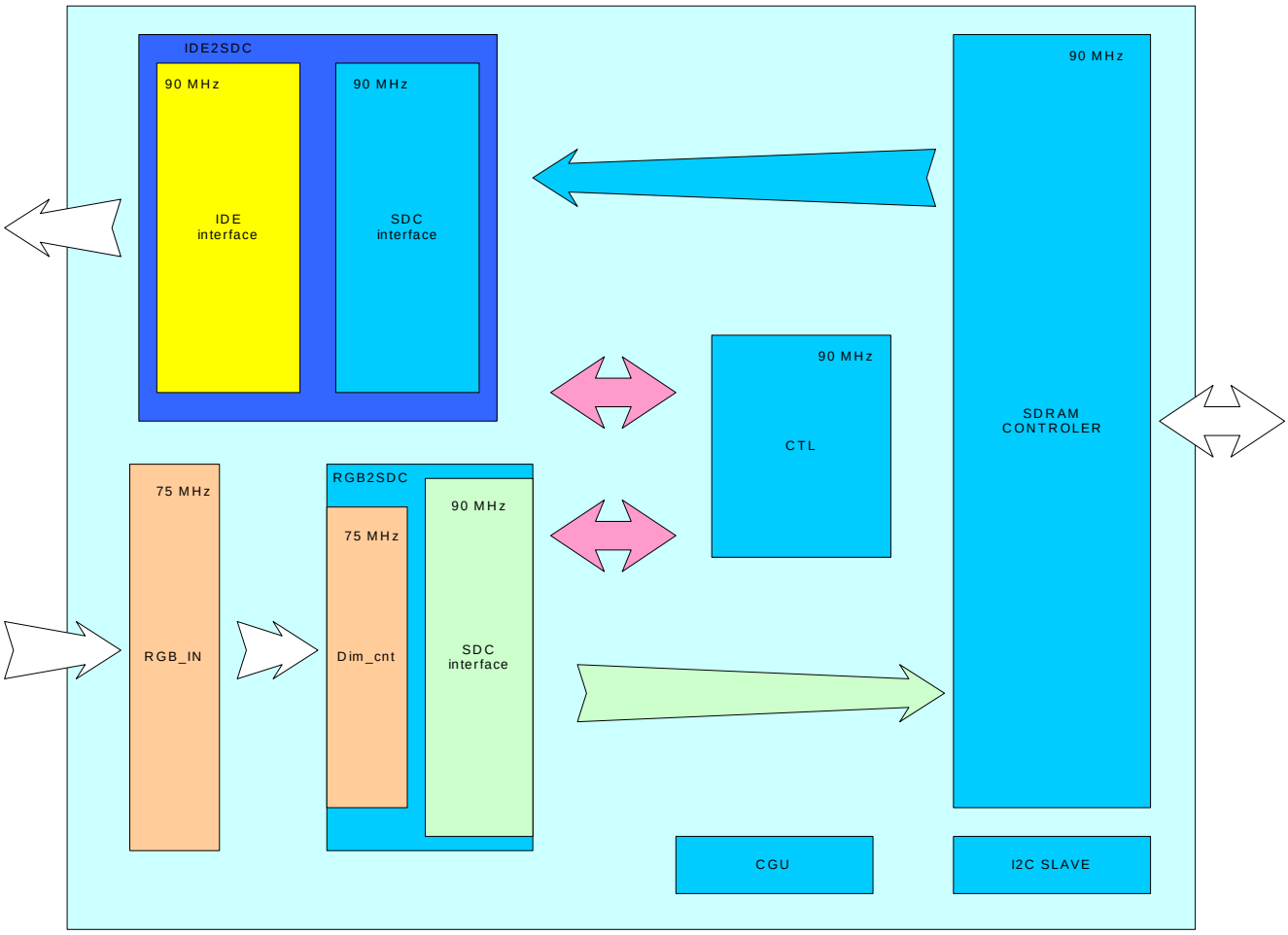
Kao memorija su iskorišćena dva SDRAM memorijska modula od 128 MB. SDRAM (*Synchronous Dynamic Random Access Memory*), ili sinhrona dinamička memorija sa slučajnim pristupom predstavlja tip dinamičke memorije kod koje se operacije čitanja, pisanja, osvežavanja, kao i druge kontrolne operacije obavljaju sinhrono sa rastućom ivicom takta na kom radi.

Povezivanje sa PC računarom ostvaruje se preko USB 2.0 sprege podržane na kontroleru. Ova sprega omogućava brzine od 12Mb/s (*Full speed*) što je sa stanovišta zahteva sistema bio dovoljan propusni opseg.

Na ploči postoji i skaler slike koji omogućava pregled obrađivane slike na PC računaru u nižoj rezoluciji u realnom vremenu.

4.IMPLEMENTACIJA U XILINX SPARTAN3

Blok dijagram FPGA projekta kao i tok signala unutar sistema implementiranog u FPGA uređaju prikazuje slika 3:



Slika 3. *Blok dijagram projekta implementiranog u FPGA*

U arhitekturi programabilno sekvencijalne mreže na slici 3. se uočavaju dva relativno nezavisna dela, ulazni (prihvatanje podataka i upis u SDRAM memoriju) i izlazni (čitanje, pakovanje i formiranje izlaznog toka). Na prelazu između ova dva dela obezbeđuje se međusobna nezavisnost ulaznih i izlaznih tokova podataka, i na taj način njihova sinhronizacija.

U nastavku je dat kratak opis svakog od implementiranih modula :

4.1 RGB_IN MODUL

Modul zadužen za registrovanje ulaznih RGB signala,kao i HSYNC i VSYNC signala sinhronizacije. Na njegove ulaze dovodi se informacija o boji, istovremeno za dva piksela, (dual mod) dva signala sinhronizacije HSYNC (horizontalna sinhronizacija) i VSYNC (vertikalna sinhronizacija), kao i signal DE (pruža informaciju o validnosti ulaznih RGB podataka). Ulazni podaci stižu na taktu RGB_CLK. Registrovani ulazi se dalje prosledjuju narednom modulu na dalju obradu.

4.2 RGB2SDC MODUL

Modul koji prihvata registovane ulazne signale na ulaznom domenu takta od 75 MHz, na osnovu njih proračunava dimenzije slike i pretvara ih u oblik pogodan za upis u memoriju. Komuniciranje sa SDRAM kontrolerom se ostvaruje taktom od 90 MHz i pomoću FIFO magazinske memorije sinhrinizuje rad na dva različita domena takta.

4.3 IDE2SDC MODUL

Uloga ovog modula je izčitavanje sadržaja SDRAM memorije i prosledjivanje putem USB porta ka PC računaru. Zbog zahteva protokola da se podaci prenose u blokovima od po 8KB, unutar ovog modula je implementirana FIFO magazinska memorija koja prihvata podatke isčitane iz SDRAM memorije i tek kada se napuni do 8KB počinje slanje datog bloka ka PC.

4.4 CTL MODUL

Ovaj modul je jedna vrsta arbitra koji na osnovu ulaznih signala odlučuje koji od dva modula RGB2SDC ili IDE2SDC će imati pristup SDRAM kontroleru, tj. da li će na snazi biti upisivanje ili čitanje iz SDRAM memorije.

4.5 CGU MODUL

Osnovni takt celom FPGA projektu obezbeđuje spoljni oscilator učestanosti 27 MHz. Za obezbeđivanje taktova od 90 MHz potrebnih u projektu, korišćeni su DCM (*Digital Clock Menagement* – digitalno upravljenje taktom) moduli koji već postoje u FPGA. Na ovaj način, korišćenjem DCM uređaja koji već postoje na samoj Spartan3 programabilnoj komponenti, uz postavljanje odgovarajućih parametara moguće je deljenjem ili množenjem ulaznog takta obezbediti ostale taktove potrebnih vrednosti.

5.PROGRAMSKA PODRŠKA

Programska podrške (slika 4.) za automatsko ispitivanje TV prijemnika obezbeđuje:

- Kreiranje postupka za ispitivanja.
- Izvršavanje postupka za ispitivanja.
- Ažuriranje postojećih postupaka za ispitivanja.
- Brisanje postupaka za ispitivanja .
- Kreiranje skupa postupaka za ispitivanja.
- Izvršavanje skupa postupaka za ispitivanja.
- Generisanje rezultata postupka za ispitivanja (HTML, EXCEL i dBASE oblik).
- Vizuelni prikaz toka postupka za ispitivanje.

5.1. KREIRANJE POSTUPKA ISPITIVANJA

Kreiranje postupaka ispitivanja se sastoji od formiranja grupe koraka, koji predstavljaju elementarne radnje koje se izvršavaju u toku ispitivanja. Korisnik u toku kreiranja postupka ispitivanja u zavisnosti od cilja ispitivanja, formira niz koraka i snima referentne uzorke sa kojima se porede uzorci preuzeti u toku izvršenja ispitivanja. Obezbeđen je rad sa sledećom grupom koraka:

- Ručna procena rezultata ispitivanja - svrha koraka je da korisnik vizuelnim putem proceni rezultat testa.
- Automatska procena rezultata ispitivanja - namenjen za automatsko poređenje slika, pri čemu se referentna slika kreira u fazi pravljenja testa.
- Vremensko kašnjenje - namenjen za stvaranje vremenskog razmaka između koraka testa.
- Upravljanje emulatorom daljinskog upravljača - namenjen za zadavanje komandi putem emulatora daljinskog upravljača.
- Upravljanje QuantumData882 generatorom signala- namenjen za kontrolu generatora video signala.
- Upravljanje Fluke54200 generatorom signala -namenjen za kontrolu generatora video i teletekst signala.
- Upravljanje HamegHM7044 izvorom napajanja - namenjen za kontrolu izvora napajanja.

5.2. IZVRŠAVANJE POSTUPKA ISPITIVANJA

Prilikom izbora postupka ispitivanja koji se pokreće odnosno pre izvršavanja postupka, korisniku su na raspolaganju podaci o svakom kreiranom postupku ispitivanja: ime ispita sa kratkim opisom namene ispita i niz pojedinačnih koraka koji se izvršavaju u toku izvršavanja ispitivanja sa opisom njihove namene u okviru ispita.

Pokretanjem izabranog postupka ispitivanja započinje se sa izvršavanjem pojedinačih koraka (elementarnih radnji), redosledom koji su definisani u procesu kreiranja ispita. U zavisnosti od tipa koraka, programska podrška na odgovara - odgovarajući način upravlja tokom procesa ispitivanja.

Moguće su 3 vrste rezultata ispitivanja:

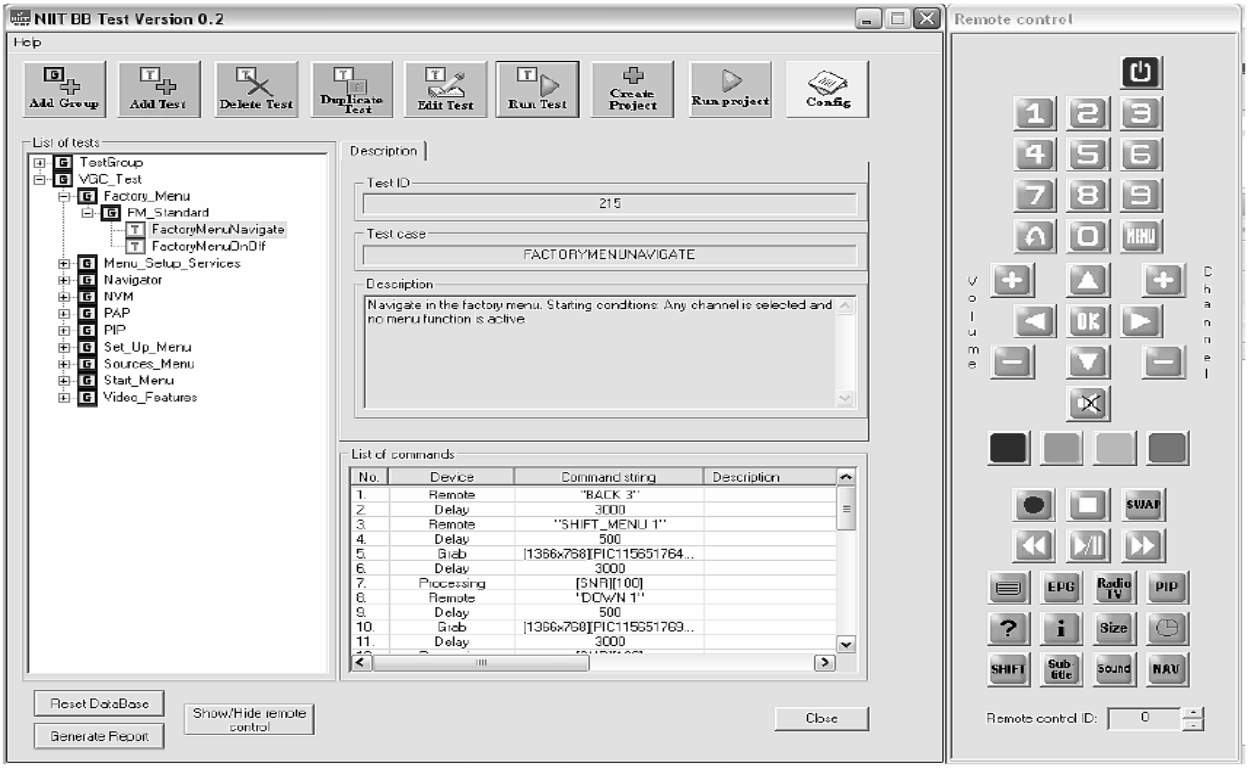
- Pass – ispitivanje je zadovoljio kriterijume .
- Fail – ispitivanje nije zadovoljilo kriterijume .
- Inccoclusive – nije moguće odrediti rezultat ispitivanja

Za potrebe ispitivanja putem većeg broja postupaka ispitivanja, koriste se projekti (grupa postupaka ispitivanja). Prilikom pokretanja projekta ispiti se izvršavaju sekvencijalno jedan za drugim, bez potrebe za ikakvom intervencijom čoveka osim ako ispit ne sadrži ručnu procenu rezultata. Ispiti se izvršavaju redosledom koji je naveden prilikom kreiranja projekta.

5.3. ARHIVIRANJE REZULTATA

Pored procedure ispitivanja uređaja, veoma važan deo procesa testiranja predstavlja način i oblik prikazivanja, odnosno saopštavanja rezultata testa. To je bitno, kako radi lakše analize rezultata ispita, tako i radi arhiviranja tj. praćenja istorije rezultata ispita. Programska podrška BBT Test generiše izveštaj rezultata testova u 4 oblika:

1. Izveštaj u obliku tekstualne datoteke (TXT format).
2. Izveštaj u obliku baze podatka (DBF format).
3. Izveštaj u obliku Internet stranice (HTML format).
4. Izveštaj u obliku Excel dokumenta (XLS format)



Slika 4.*Izgled programske podrške*

6.ZAKLJUČAK

Ovaj sistem je namenjen automatskom ispitivanju TV prijemnika, pri čemu je akcenat stavljen na što jednostavniju arhitekturu sistema i što nižu cenu izrade, a pritom da bude zadovoljena neophodna funkcijonalnost.

7.LITERATURA

[1] Vladimir Kovačević : *Logičko projektovanje Računarskih sistema,*Univerzitet u Novom Sadu, Fakultet Tehničkih Nauka, 1993

[2] Application note: Spartan-3 FPGA Family : Complete Data Sheet www.xilinx.com

[3] Application note: THC63LVD1024_Rev1.0_E1 www.thine.co.jp

[4] Application note: Synchronous Dram MT48LCM32B2 www.micron.com

Abstract – This paper describes one solution of hardware arhitecture based on FPGA and PC application which is used for automatic testing of TV.

AN EXAMPLE OF SYSTEM FOR AUTOMATIC TV TESTING BASED ON FPGA

Aleksandar Tomašević, Zoran Krajačević,Miloš Nikolić

PREGLED SMART GRID KONCEPTA I OSVRT NA PRIMENU U AMERIČKIM MREŽAMA

Dejan Grabež, Mirna Galić, Dario Trivunović, Predrag Vidović, *Fakultet tehničkih nauka, Novi Sad*

Sadržaj – *U radu je izložena problematika današnjeg elektroenergetskog sistema, koja sa ubrzanim porastom dimenzija istog, dovodi u pitanje buduću funkcionalnost postojeće opreme. Prikazan je sam koncept Inteligentnih mreža sa nivoima inteligencije i izdvojeni su primeri nekih elemenata čije se uvođenje zahteva u ostvarenju cilja – mreže koja će biti u mogućnosti da zadovolji buduće zahteve potrošača.*

1. UVOD

Elektroenergetski sistem (EES) – proizvodnja, prenos, distribucija i potrošnja, idejno je izgrađen pre više desetina godina. Ovaj energetski lanac je specifičan, jer napajanje i zahtevi moraju biti konstantno usko balansirani, jer ne postoji komercijalno rešenje za skladištenje električne energije u cilju absorbovanja narušene proizvodno/potrošačke ravnoteže.

U prošlosti, ravnoteža je ostvarivana unutar vertikalne integracije preduzeća koja je kontrolisala i proizvodni i sistem isporuke. Današnja mreža je pred mnogo većim izazovom: deregulacija, koja je dovela do trgovanja električnom energijom unapred zadajući tokove snaga i neizvesnosti za koje sistem nije projektovan; proboj obnovljivih izvora električne energije koji pomeraju postojeću infrastrukturu proizvodnje, preopterećenost sistema zbog sve većih zahteva, što zajedno sa neinteligentnim menadžmentom i zastarelom opremom dovodi do potpunih ispada sistema (ispad u SAD 14.08.2003.); zahtev za redukovanjem zagađenja okoline – prema istraživanjima američke nacionalne laboratorije za obnovljive izvore, današnja emisija od 1700 miliona tona ugljenika godišnje (Mtc), porašće do 2030. na 2300 (Mtc) itd.

Generalno, raste i zahtev za električnom energijom, gde se predviđa do 2050. godine potrošnja od 30-60 TW električne energije (oko 20TW samo u SAD).

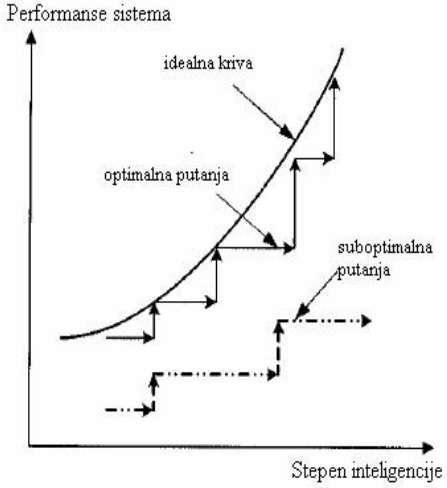
U SAD-u (i Evropi) su čitava energetska industrija, vlada, istraživački centri i različite kompanije pokrenuli inicijativu sa istim ciljem – kako omogućiti da se prevaziđu svi navedeni tehnički, ekološki i sigurnosni problemi koji prelaze granice kontrole, funkcionalnosti, zaštite i planiranja EES-a. Optimizacija problema može biti postignuta povećanjem inteligencije energetskog lanca. [1, 3, 5, 7]

2. INTELIGENTNE MREŽE

Primena Smart Grid koncepta (poznatijeg u SAD kao *Intelligent Grid* - inteligentna mreža) je fokusirana na uvođenje određenog nivoa inteligencije u svaki podsistem EES-a, u svaki njegov element i u njegov doprinos poboljšanju performansi sistema.

Na slici 1 prikazane su dve moguće razvojne putanje u smislu razvoja inteligencije i prostora za performanse.

Prva putanja (suboptimalna putanja) predstavlja situaciju u kojoj je svaki pojedinačni element sistema ekstremno inteligentan i optimizovan za sopstvene performanse, ali sveukupne performanse EES-a nisu poboljšane. Investicije u podesan stepen inteligencije će dovesti do željenih poboljšanja u performansama sistema.



Slika 1 – Razvojne putanje

Druga putanja (optimalna putanja) predstavlja situaciju gde je nivo inteligencije različitih komponenti sistema poboljšan u skladu sa traženim zahtevima, tj. svaki stepen inteligencije je bliži idealnoj krivi koja predstavlja maksimizovan odnos dobit/troškovi.

Sledeća lista identifikuje nivoe inteligencije:

Nulti nivo – neinteligentna oprema, kao što su prenosni vodovi i kablovi, kondenzatori, izolatori itd.

Prvi nivo – komponente koje imaju lokalno očitavanje podataka sa ili bez lokalnih akcija. Primeri inteligentnih komponenti na ovom nivou su releji, regulacione sklopke transformatora, digitalni detektori kvara, naponski regulatori,...

Drugi nivo – razmena podataka između komponenti prvog nivoa ("master-slave" razmena ili razmena "peer to peer") koje sarađuju sa komponentama iz okruženja. Primeri su distributivni fideri, automatski merači, transformatorske stanice.

Treći nivo – razmena informacija između različitih okruženja koja može imati za rezultat promenu nekih akcija na drugom nivou, npr. razmena podataka između više transformatorskih stanica.

Četvrti nivo – razmena informacija između drugog nivoa i regionalnih kontrolnih jedinica. Ove jedinice su odgovorne za koordinaciju svih akcija na drugim nivoima. Ovde spadaju *EMS* (*Energy Management System*), *SCADA* (*Supervisory Control and Data Acquisition*) i *DMS* (*Distribution Management System*) kontrolni centri.

Peti nivo – razmena informacija između kontrolnih centara (centri na četvrtom nivou) radi koordinacije akcija regionalnih kontrolnih centara i njihove saradnje. [4, 5]

Napomene: a) Rad je proistekao iz diplomskog-master rada Dejana Grabeža. Mentor je bio prof.dr Dragan Popović.
b) Rad je prethodno publikovan an konferenciji INDEL 2008, Banja Luka, Novembar 2008.

3. OSNOVNE RAZLIKE DANAŠNJE U ODNOSU NA INTELIGENTNE MREŽE BUDUĆNOSTI UZ NEKE OSTVARENE PRIMERE POVIŠENJA INTELIGENCIJE

Mreža danas je pretežno elektromehanička, sa senzorima u pojedinim delovima. Restauracija je ručna, zasnovana na ljudskom faktoru, uz izuzetke preduzeća koja su već počela sa ugradnjom DMS sistema (EMS sistema), čime se postiže efikasnija kontrola distributivnih mreža (prenosnih mreža), planiranje, kao i nadzor uz veliki set funkcija, od kojih je jedna i restauracija napajanja. Generatori su centralizovani, provera opreme je i dalje pretežno ručna, postoji samo jednosmerni protok informacija. Sve bitne odluke donose stručnjaci i nije ustaljena informacija o tarifnom sistemu, kao i izbor tarife.
Inteligentne mreže biće digitalizovane, sa dvosmernim protokom informacija. Stručnjacima će mnogi softverski alati pomagati u donošenju odluka, mreža će imati mogućnost inteligentnog nadzora. Potrošači će imati uvid u tarifni sistem i mogućnost biranja tarife prema potrebama. Biće uvedeni distributivni generatori i mogućnost daljinskog nadzora opreme, sa mogućnošću međusobnog komuniciranja između pojedine opreme. U narednom delu poglavlja biće dati primeri povišenja inteligencije EES-a.

3.1. Visoko temperaturni superprovodni (high temperature superconducting - HTS) strujni limiteri za zaštitu mreža od strujnih udara

Struje kvara, koje su znatno veće od nominalnih struja elemenata sistema, uzrok su ispada vodova, nedozvoljenih padova napona, itd. Na taj način degradiraju kvalitet električne energije. Kako elektroenergetski sistemi rastu, raste i njihova povezanost, kao i učestanost kvarova (kratkih spojeva). Struje kvara mogu oštetiti skupu opremu i izazvati gubitak važnih informacija, što prouzrokuje visoku materijalnu štetu. Upotreba računara i drugih mikroprocesora čija izdržljivost na strujne udare opada sa povećanjem njihove osetljivosti mogu da pogoršaju situaciju ako dođe do njihovog kvara. Standardan način za otklanjanje struje kvara putem rekonfiguracije mreže ima visoku cenu, a ujedno smanjuje fleksibilnost i pouzdanost sistema. Primena rekonfiguracije može biti zadovoljavajuća ukoliko se koristi softversko rešenje u okviru DMS paketa.

Uređaji poznati pod nazivom "*fault current limiter*" (*FCL*) tj. limiteri struje kvara, mogu da ograniče struju kvara i na taj način zaštite sistem. Do nedavno FCL-ovi su bili ili jako skupi ili su im performanse bile loše. Uzrok tome bilo je nepostojanje materijala koji bi imao potrebne osobine, kao i nemogućnost realizacije dobrog hlađenja. Fundamentalni princip ograničenja i kontrolisanja struje kvara je stavljanje promenljive impedanse u seriju sa štićenim elementima elektroenergetskog sistema – Ohm-ov zakon. FCL ograničava neželjeni strujni talas tako što povećava impedansu onog trenutka kada se pojave uslovi struje kvara. Idealni FCL bi trebao posedovati sledeće karakteristike:

- nultu ili jako malu impedansu za normalne uslove rada, sa veoma malim gubicima energije;
- veoma veliku impedansu usled uslova struje kvara, dovoljnu da ograniči strujni talas tako da zaštiti sistem i opremu;
- kratko vreme povratka u normalan režim rada nakon otklanjanja struje kvara, da bi se obezbedila što brža restauracija napajanja;
- visoku pouzdanost tokom dugog vremenskog perioda;
- kompaktan i jednostavan dizajn kao i malu težinu;
- nisku cenu uređaja i njegove eksploatacije.

Generalno postoje tri vrste FCL-a u zavisnosti od prirode impedanse. To su:

- čisto omski;
- kombinovani omsko-induktivni;
- čisto induktivni;

Većina koncepta FCL-ova je još uvek samo "na papiru" koji tek treba da se proizvedu i prođu testiranje. Najnapredniji realizovani koncepti su HTS FCL-ovi, a kompanija koja je uspešno razvila i testirala jednofazni 6.4 MVA superprovodni strujni limiter (SCFCL) je švajcarska kompanija ABB. [2, 5]

3.2. Visoko temperaturni superprovodnik/super tečni vodonik – energetski put za prenos velike količine "čiste" energije potrebne za podmirenje potreba za energijom u budućnosti

Superprovodnost je osobina klase provodnika pod nazivom superprovodnici, čija otpornost postaje jednaka nuli kada se provodnik ohladi na temperaturu manju od prelazne temperature (*Tc*.). Superprovodnik gubi neke od svojih osobina kada gustina struje kroz njega premaši kritičnu vrednost (*Jc*) i tada je magnetno polje koje potiče od superprovodnika veće od dozvoljene vrednosti (*Hc*.).

Značaj superprovodnika u elektroenergetici je prepoznat odmah nakon njihovog otkrića 1911. godine. Nažalost, superprovodnici koji su tada otkriveni nisu imali značaja za elektrotehniku jer im je prelazna temperature iznosila čak 23 K (-418 F tj. -250 ^{o}C). Godine 1987. otkriveni su visoko temperaturni superprovodnici (*HTS*) sa prelaznom temperaturom od 77 K (-196 ^{o}C), što je drastično promenilo situaciju i mogućnost realizacije prenosa električne energije sa zanemarljivo malim gubicima.

U proteklih 15 godina otkriveni su mnogi HTS provodnici. Vremenom, prelazna temperatura je dostigla 134 K (-139 ^{o}C), a pod visokim pritiskom čak 164 K (-109 ^{o}C). Koristeći BSSCO (Bi2Sr2Ca2Cu3O8) masu, proizvođači su uspeli da naprave 1 kilometar dužine HTS provodnika. U proteklih 12 meseci SAD, Japan i Nemačka su postigli izuzetne rezultate koristeći YBCO masu (Yba2Cu3O7).

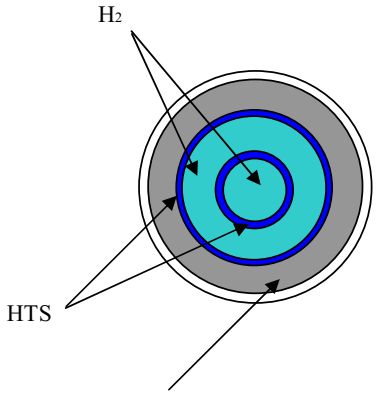
Tečni vodonik koji bi se koristio za hlađenje prenosnih vodova (da bi se postigle njihove najbolje performanse) proizvodio bi se iz fosilnih goriva, a ujedno bi se mogao koristiti za pogon motornih vozila. [5]

Poprečni presek superprovodnika sa tečnim vodonikom prikazan je na slici 2.

3.3. Prvi američki nanogrid – kvantni prenosni vod za nacionalni smart grid osnovni test

Cilj stvaranja kvantnih vodova je da se omogući prenos snage reda TW (1.000.000 MW) sa jednog kraja američkog kontinenta na drugi. Ovaj novi tip prenosnih vodova omogućava 3 do 5 puta veći prenos snage od bakarnih provodnika. Keramički presvučen superprovodnik,

razvijen od strane Los Almos-a, za manje od dve godine će se proizvoditi po ceni veoma bliskoj ceni bakarnih provodnika.



Slika 2 – Poprečni presek provodnika.

Ovaj provodnik, ohlađen tečnim azotom na -320 F, prenosi struju bez otpornosti – zanemarljiva je. Superprovodnik bez otpornosti može da prenosi veću struju od običnog bakarnog provodnika. Ne tako davno, proizvođači provodnika (American Superconductor Inc. In Westborough, Mass, and SuperPower Inc. In Schenectady) uspeli su da naprave superprovodnik dužine svega 10 metara. Oni očekuju da će za oko dve godine uspeti da proizvedu superprovodnik dužine 1 kilometar.

Superprovodni kablovi, koji moraju biti hladni, mogu zameniti bakarne samo ako su energetski vodovi pokopani. Cena ukopavanja energetskih vodova čini ovu tehnologiju prihvatljivom samo u gusto naseljenim predelima.

Nova nano tehnologija koja će biti veoma važna tek nakon pet godina odnosi se na:

- kablove za velike struje koji će zameniti postojeću električnu mrežu;
- baterije i super kondenzatore koji će poboljšati mogućnosti distributivnih generatora od 10 do 100 puta;
- nanoelektroniku koja će zameniti senzore i energetske kontrolne uređaje;
- skladišta-revolucija lakih metala za rezervoare visokog pritiska, zamajce i pretvaranje vodonika;
- gorivne ćelije distributivnih generatora – doći će do pada njihove cene od 10 do 100 puta;
- svetla nanotehnologije za zamenu sijalica sa užarenim vlaknom fluorescentnim sijalicama.

Dakle, nano provodnici su dosta superiorniji od bakarnih jer pored toga što im je električna provodnost veća, oni su lakši, otporniji na visoke temperature, imaju veću čvrstoću itd. [5].

3.4. Simulacione mogućnosti modelovanja prenosnih i distributivnih zagušenja, kaskadnih ispada, raznih nepogoda i ponovnog uspostavljanja rada

DSTS (Decision Support Threat Simulator – Simulator nepogoda u mreži za pomoć pri donošenju odluka) /*DEW (Distribution Engineering Workstation* – Radna stanica za distributivni inženjering) je sistem dizajniran da ubrzano uči iz mrežnih simulacija različitih scenarija zagušenja i kaskadnih ispada, usled delovanja vremenskih nepogoda, ispada opreme, terorizma i ostalih slučajeva sa kojima bi sistem operatori morali da nauče da se izbore u sistemu budućnosti. DSTS sistem učenja koristi DEW simulator da bi došao do odgovarajućih reakcija u upravljanju nepogodama. DSTS identifikuje i optimizira odgovore na sledeće probleme koji pogađaju mrežu:

- Koje su nove pretnje elektroenergetskom sistemu?
- Koje su nove kritične tačke u mreži?
- Koje su nove šeme nepovoljnih razvoja događaja i kaskadnih efekata?
- Koji su najbolji odgovori na date scenarije?

DSTS sistem učenja je inicijalizovan sa matricama učenja koje predstavljaju generalizovan oblik veštačkih neuralnih mreža. Dve ove matrice spregnute zajedno predstavljaju dva nivoa neuralne mreže. Tokom procesa učenja menjaju se težinski faktori ovih matrica pod uticajem simulatora i reakcija operatora. [5]

3.5. Inteligentne podstanice dodate sa ciljem preusmeravanja toka snage oko zagušenja u mreži i preuzimanja akcija predodređenih simulacijama

Svaki inteligentni kontroler dodat sistemu prenosa za preusmeravanje opterećenja oko zagušenja bi eliminisao potrebu za oko 100 MW buduće proizvedene snage. Problem preusmeravanja toka snage postao je jednako značajan kao i problem nedostatka proizvodnih kapaciteta. Električna energija se prenosi na sve veće udaljenosti, što dodatno komplikuje dati problem. Javlja se potreba za prekidačima koji su dovoljno jaki i brzi da kontrolišu tok snage, gde dolaze do izražaja tiristori snage. Interkonektivni vodovi su prvobitno uvedeni sa ciljem povećanja pouzdanosti i smanjenja troškova, međutim njima se prenose i poremećaji. Oni se šire mrežom nelinearno, izbacujući podstanice u talasima. Sa višim naponima dolazi i veća dinamička nestabilnost, tako da su operatori često prinuđeni da ograničavaju opterećenje vodova na 60% njihovog termičkog kapaciteta. Kombinacija energetske elektronike sa Smart Control sistemima – inteligentnim sistemima za kontrolu, može da nadomesti ovaj gubitak kapaciteta koristeći programabilne procesore koji koriguju porast ili pad napona u delićima sekunde, prepodešavanjem teretnih menjača transformatora ili promenom topologije mreže. *FACTS* (*Flexible AC Transmission System*-fleksibilni AC sistem prenosa) omogućava skoro trenutnu kontrolu tokova snaga, čime stabiliše sistem posle poremećaja, olakšava ili čak eliminiše zagušenja, povezuje mreže, integriše distributivne generatore, onemogućava stvaranje kružnih struja. Par statičkih sinhronih kompenzatora menjaju oblik naizmenične struje voda, čime omogućuju prenos energije sa jednog voda na drugi. Električna energija je skladištena i oslobođena pomoću kondenzatora velikog kapaciteta. [5]

3.6. Dodavanje distributivnih generatora i distributivnog skladištenja električne energije

Unapređenje efikasnosti rada sistema utiče kako na proizvodnju, tako i na potrošnju električne energije.

Industrijskim potrošačima mora se omogućiti da učestvuju u funkciji odsecanja opterećenja, ako postoji mogućnost njihovog napajanja iz distributivnog generatora. Time bi se smanjilo vršno opterećenje, bez gubitka produktivnosti. Tehnički problemi izjednačenja napona, faznog stava i učestanosti distributivnog generatora i mreže je problem koji inteligentna mreža može da reši. Upotreba DC vodova može da pruži značajno olakšanje takvog problema. Distributivni generatori imaju nižu termičku efikasnost i veću emisiju štetnih gasova, dok centralizovane elektrane, ukoliko su značajno udaljene od mesta potrošnje, imaju velike gubitke električne energije u prenosu. Distributivni generatori i skladišta trebaju biti postavljeni na udaljene krajeve distributivne mreže koji su posebno ranjivi na kolebljivost električne energije. Tehnički problemi povezivanja distributivnih generatora su:

- Može li se povezivanje distributivnih generatora sa krajnjim korisnicima učiniti jednostavnijim?
- Može li se značajan broj distributivnih generatora povezati i u upetljane i u radijalne mreže?
- Da li postoji sigurno, pouzdano i troškovno efikasno povezivanje distributivnih generatora?
- Da li se rešenja povezivanja mogu izvršiti pravovremeno?
- Da li se inženjerske studije za povezivanje mogu eliminisati, standardizovati ili automatizovati?
- Kako utvrditi da li je dati distributivni generator kompatibilan sa opremom krajnjeg korisnika ili sa drugim distributivnim generatorima?
- Da li se kvalifikovani sistemi povezivanja mogu sertifikovati tako da budu instalirani uz minimum testiranja na terenu?

Tehnologija skladištenja energije omogućava korišćenje električne energije tokom kratkog vremenskog perioda. Ona je od suštinske važnosti za korekciju padova, porasta i podrhtavanja napona, koja se javljaju prilikom uključenja distributivnih generatora i potrošača. Primeri tehnologije razvoja skladištenja električne energije su:

- Baterije;
- Konvencionalne i superprovodne rotirajuće mase;
- Superprovodna skladišta magnetne energije;
- Superkondenzatori;

Skladišta električne energije koja koriste vazduh pod pritiskom. [4, 5]

4. ULOGA SOFTVERA U OSTVARENJU CILJA – STVARANJU INTELIGENTNE MREŽE

Kako je već naglašeno, treba postići potpunu digitalizaciju mreže kao celine. Danas SCADA sistemi omogućavaju daljinski nadzor elemenata, ali samo na višim naponskim nivoima. Ovo dovodi do problema nemogućnosti pristupa elementima na niskonaponskim delovima, odnosno do pristupa vrednostima električnih veličina na samim elementima. Planerima distributivne mreže, kao i dispečerima, trebaju tačne, brze i detaljne informacije kako o trenutnom stanju, tako i o stanju u prošlosti. Softverska rešenja za date probleme postoje, poput DMS-a za distributivne mreže, gde se pomoću isprogramiranih funkcija, primenom različitih algoritama i estimacije stanja, dobijaju željeni rezultati i informacije i za niže napone. Međutim, ovaj softverski paket nije rasprostranjen.

Fundamentalna komponenta stvaranja inteligentnih mreža biće robusna i dinamična mreža komunikacija. Ona će obezbediti navedene brze (real-time) i tačne podatke, dvosmernu komunikaciju i interakciju svake komponente sistema, od proizvodnje do krajnjih potrošača. Potrebni su naravno i novi fizički uređaji (npr. prekidači, osigurači, kao i merni uređaji na potrošačkoj strani). I na kraju treba obezbediti logičku integraciju svih informacija. Za sve navedeno ključan je softver, koji omogućava mrežu komunikacija, ali i menadžment nad dobijenim podacima. [6, 7]

5. ZAKLJUČAK

Ukoliko se želi ostvariti stabilan i pouzdan sistem budućnosti sa sigurnim i neprekidnim napajanjem potrošača, koji moderno tehničko društvo zahteva, investicije će biti velike, trebaće dug vremenski period, ali će inteligentna mreža u potpunosti podržati ekonomski razvoj i zahteve nametnute većom potrebom za električnom energijom. Viši nivo inteligencije zahteva razvoj i širu upotrebu naprednih tehnologija, koje bi omogućile digitalno i daljinsko upravljanje sistemom i komunikaciju među elementima. Takođe, povećanje inteligencije dovodi do novih funkcija koje poboljšavaju upravljanje EES-a. [3]

6. LITERATURA

[1] http://www.pewclimate.org/docUploads/
10-50_Anderson_120604_120713.pdf
From workshop proceedings, "The 10-50 Solution: Technologies and Policies for a Low-Carbon Future."The Pew Center on Global Climate Change and the National Commision on Energy Policy. naziv rada The Ditributed Storage-Generation "Smart" Electric Grid of the Future

[2] http://www.w2agz.com – 6.4 MVA resitive fault current limiter based on Bi-2212 superconductor

[3] S. Massoud Amin, Bruce F. Wollenberg, "Toward a Smart Grid", IEEE power & energy magazine, september/october 2005.

[4] Standardizing the Classification of Intelligence Levels and Performance Of Electricity Supply Chains
NEMA
Contributors: ABB, Cooper Power Systems, Eaton, Emerson, GE, Nexans, Rockwell Automation, Siemens, Square D, Thomas and Betts 12-18-2007

[5] http://www.ldeo.columbia.edu/res/pi/4d4/testbeds/Smart-Grid-White-Paper.pdf

[6] EPRI Smart Grid Demonstrations Overview – Mark McGranaghan Jun 2008

[7] http://www.xcelenergy.com/docs/
SmartGridWhitePaper.pdf

Abstract – The paper deals with a problem of today's power systems and it's fast growth. This leads to a question of future functionality of its equipment. The concept of Smart Grid is given with its levels of intelligence and some examples of necessary future elements.

SMART GRID CONCEPT WITH AN OVERLOOK ON IMPLEMENTATION IN AMERICAN GRID

D. Grabež, M. Galić, D. Trivunović, P. Vidović

JEDNO REŠENJE SISTEMA ZA KONTROLU CMOS SENZORA

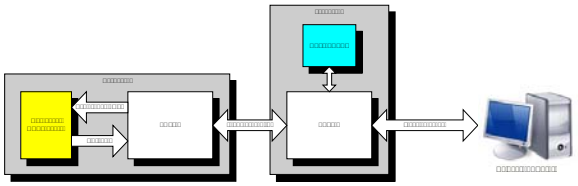
Dragan Topalović, Dušan Majstorović, Zoltan Pele, Predrag Eremić, *Fakultet tehničkih nauka, Odsek za računarsku tehniku i računarske komunikacije*

Sadržaj – *Rad opisuje jedno rešenje realizacije sistema zasnovanog na programabilnoj sekvencijanlnoj mreži (FPGA). Sistem predstavlja kameru visoke definicije i velike brzine. U okviru rada detaljno je opisan podsistem zadužen za kontrolu i upravljanje prenosom podataka sa komercijalno dostupnog sensora, kao i prilagođenje formata podataka za slanje preko serijske sprege visoke propusne moći.*

1. UVOD

U savremenim naučnim istraživanjima koja uključuju kretanje virusa, snimanje veličina teško uočljivih ljudskim okom, brze promene posmatranog sistema i sl. javlja se potreba za kamerama visoke propusne moći. Od kamera ovog tipa često se zahteva podrška za neprekinut prenos podataka prema korisniku u smanjenoj rezoluciji i sa smanjenom učestanošću uzorkovanja. Ovaj režim rada se najčešće koristi za podešavanje aparature i pravilno pozicioniranje kamere. Međutim, za vreme samog eksperimenta, kada kamera radi sa punom rezolucijom i punom učestanošću neprekinut prenos nije moguć usled prevelikog zahtevanog protoka podataka. U ovom slučaju, podaci se smeštaju u kružni bafer i isključivo na zahtev korisnika, određena sekvenca se prenosi na računar.

Slika 1 prikazuje fizičku arhitekturu sistema koji obezbeđuje podršku za digitalni CMOS senzor visoke definicije (*MAPS - MegaPixel CMOS Active-Pixel digital Image Sensor*). Ovakav sistem predstavlja kameru visoke propusne moći sa maksimalnom rezolucijom od 1.3MP i učestanošću od 500FPS. MAPS sistem se sastoji od dve odvojene štampane polože: prednje ploče (*Headboard*) i osnovne ploče (*Baseboard*).



Slika 1 - Fizička arhitektura sistema

Za realizaciju kamere je izabran Micron MI-MV13 senzor. Kontroler zadužen za upravljanje senzorom je realizovan unutar programabilne sekvencijalne mreže (*FPGA - Field Programming Gate Array*). Težište ovog rada je na opisu fizičke arhitekture realizovane unutar programabilne komponente na prednjoj ploči (FPGA 1 na Slika 1).

Unutar FPGA komponente realizovane su sledeće funkcije:

- Upravljanje procesom akvizicije slike na samom senzoru
- Upravljanje DAC-ovima koji obezbeđuju analogne nivoe neophodne za rad analogne sekcije senzora
- Obezbeđen je stabilan rad sistema podeljenog na više domena takta
- Podrška za akviziciju podataka sa senzora na dva načina rada (*preview* i *capture*)
- Mogućnost promene učestanosti uzorkovanja slike (*frame-rate*), kao i dela slike koji se prenosi
- Prijem upravljačkih komandi preko serijskog sprežnog sistema kao i prosleđivanje statusnih informacija ka ostatku sistema

2. FIZIČKA ARHITEKTURA SISTEMA

Senzor se nalazi na prednjoj ploči koja je sa glavnom pločom povezana visoko propusnom vezom. Prednja ploča će se nalaziti na pokretnoj aparaturi unutar izolovane prostorije prilagođene uslovima rada istraživanja. Osnovna ploča se nalazi izvan izolovane prostorije. Maksimalno projektovano rastojanje između ploča je 50 cm, a između osnovne ploče i PC je maksimalno 20m.

Prema zahtevima sistema i proceni potrebnih hardverskih resursa izabrana je Xilinx *Virtex4 XC4VFX20-11FF672C* programabilna sekvencijalna mreža.

Virtex4 je programabilna sekvencijalna mreža sa konfigurabilnim elementima i ugrađenim jezgrima optimizovanim za rad na visokim učestanostima.

Ulazmo/izlazni (I/O) blokovi obezbeduju spregu izmedu spoljnih priključaka i interne konfigurabilne logike. Konfigurabilni logički blokovi (*CLB - Configurable Logic Blocks*) su osnovni elementi u Xilinx FPGA komponenti. Obezbeđuju podršku za kombinacionu i sinhronu logiku, distribuiranu memoriju i pomeračke registre. Ugrađeni memorijski blokovi obezbeđuju fleksibilnu 18Kbit dvoprilaznu RAM koji mogu biti organizovani u kaskadni sistem. Memorijski blokovi *Virtex4* komponente imaju opcionu programabilnu magacinsku memoriju koja dodatno povećava fleksibilnost komponente. Kaskadni ugradeni namenski blokovi za podršku digitalnoj obradi signala sadrže 18x18 bitne množače, sabirače i 48 bitne akumulatore.

Virtex4 komponenta podržava sledeću ugrađenu funkcionalnost:

- Integrisane primopredajnike velike brzine (*MGT - Multi Gigabit Transceiver*) koji ostvaruju brzine i do 6.5 Gb/s po kanalu
- Integrisani IBM *PowerPC 405* RISC CPU sa maksimalnom podržanom učestanošću rada od 450 MHz
- Ethernet sprežni sistem sa podrškom za brzine od 10/100/1000 Mbita (*EMAC - media-access control*)

Napomene:

a) Rad je proistekao iz diplomskog-master rada Dragana Topalovića. Mentor je bio prof.dr Nikola Teslić.

b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

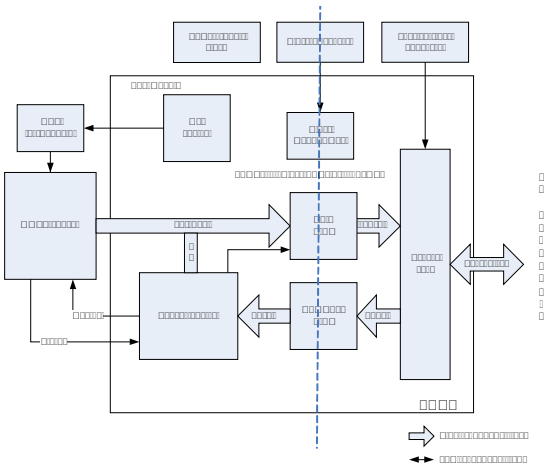
Virtex4 familija Xilinx-a znatno povećava mogućnosti programabilnih sekvencijalnih mreža, predstavljajući veoma značajnu alternativu ASIC tehnologiji. *Virtex4* uključuje *PowerPC* processor, Ethernet jezgro sa podrškom za tri standardne brzine rada kao i primoredajnike sposobne za prenos podataka učestanošću od 622 Mb/s do 6.5 Gb/s. Tabela 1 prikazuje raspoložive resurse izabrane programabilne komponente.

raspoloživi resursi programabilne komponente	XC4V FX20
broj logičkih ćelija	23,040
maksimalan broj distribuiranih RAM blokova	134
broj 18 Kb blokova	68
kapacitet blok memorije (Kb)	1,224
broj jezgara za kontrolu takta (*DCM*)	4
broj *PowerPC* procesora	1
broj jezgara za mrežnu podršku (*EthernetMAC*)	2
broj primopredajnika velike brzine (*MGT*)	8

Tabela 1 - Karakteristike Virtex4 integrisane komponente

Slika 2 predstavlja blok šemu FPGA strukture na prednjoj ploči. Projektovani sistem se sastoji od sledećih modula:

- Sensor kontroler
- DAC kontroler
- Podsistem za generisanje i razvođenje takta
- FIFO memorije podataka i komandi
- Aurora sprežni sistem

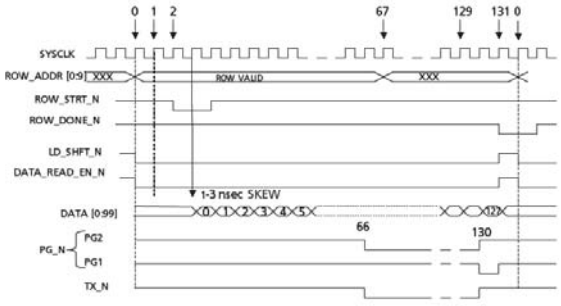


Slika 2 - Blok šema prednje ploče

3. SENZOR

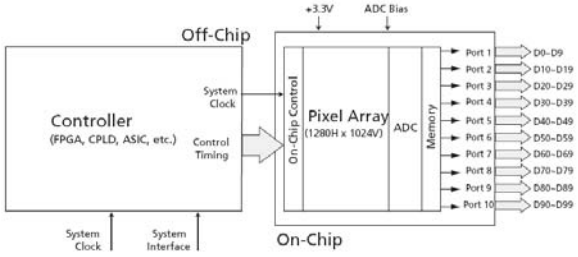
Od dostupnih komercijalnih sensora izabran je MI-MV13 je 1,280H x 1,024V (1.3 megapixel) CMOS digitalni senzor propusnosti 500 slika u sekundi (*fps - frames-persecond*). Senzor je dostupan u monohromatskoj i višebojnoj verziji. Unutar samog senzor integrisanog kola nalazi se 1280 desetobitnih analogno-digitalnih pretvarača (*ADC - Analog To Digital Converters*), odnodsno po jedan za svaku kolonu, čime je omogućena potpuno digitalna sprega sa korisnikom (Slika 4). Pri učestanosti odabiranja 60 slika u sekundi senzor troši manje od 150mW, a pri maksimalnoj brzini od 500 manje od 500 W, pri radnom naponu od 3.3V. Površina piksela je 0.012mm².

Slika 3 predstavlja vremenski dijagram upravljačkih signala senzora prilikom prenosa podataka jednog reda slike. Nakon postavljanja kontolnih signala i prosleđivanja validne adrese reda, potrebno je pravilno preneti podatke sa senzora, kao i kontrolnim signalima obavestiti senzor o uspešnom prenosu kako bi se inicirao prenos sledećeg reda.



Slika 3 - Vremenski dijagram kontrole pri prenosu jednog reda slike

Za pravilan rad analogne sekcije senzora neophodno je obezbediti potrebne analogne nivoe na odgovarajućim ulazima. Za generisnje potrebnih naponskih nivoa korišteni su D/A pretvarači AD5327BRUZ firme *Analog Devices*[4].



Slika 4 - Kamera sistem realizovan pomoću MI-MV13 CMOS digitalnog senzora

4. UPRAVLJAČ SENZOROM

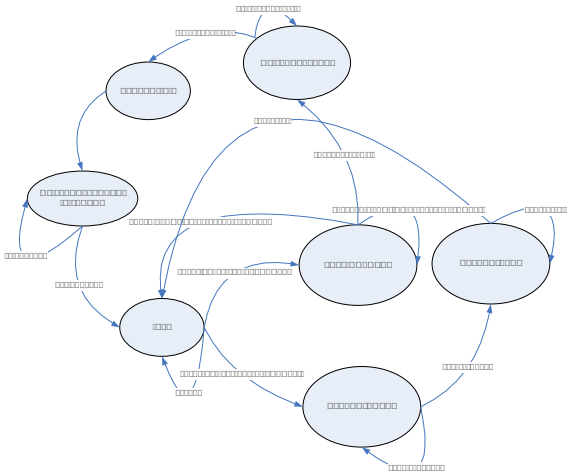
Predstavlja modul odogvoran za upravljanje senzorom. Prima upravljačke komande na osnovu kojih upravlja senzorom, takođe obezbeđuje statusne informacije i prosleđuje ih preko osnovne ploče korisniku (PC aplikaciji).

Modul je realizovan iz dva osnovna dela:

- Prilagodnog dela koji podešava osnovne parametre sistema (broj slika u sekundi, definiciju dela slike koja se prenosi, podešavanja parametara senzora) i osnovne komande kao što su: restartovanje, pokretanje sistema, početak prenosa, broj slika koji se prenosi nakon iniciranja prenosa. Takođe,

potrebno je obezbediti statusne informacije o trenutnom stanju automata, o aktuelnom načinu rada. Prilikom prenosa podataka (*capture stream*) potrebno je informisati korisnika o poziciji tekuće slike (npr. 30 slika od mogućih 50), o poziciji unutar same slike (informacije o koloni i red) i sl.

- Dijagram automata s konačnim brojem stanja prikazan je na slici Slika 5. Automat generiše pravilnu sekvencu upravljanja i obezbeđuje prenos podataka sa senzora. Ulazi automata se preko sprežnog sistema i komandne FIFO memorije prenose od korisnika, kao što su zahtev za prelazak u željeni način rada, ponovno pokretanje sistema i sl.



Slika 5 - Automat s konačnim brojem stanja

Realizovano rešenje kontrolera podržava dva načina rada:

- Osnovni način rada sistema je konstantno prikazivanje slika sa senzora (*preview mode*), pri smanjenoj rezoluciji slike 640x512, kao i pri smanjenom broju slika u sekundi, reda 30. Na taj način korisnik stiče uvid o posmatranom događaju, može da izvrši dodatna podešavanja, da definše deo slike od interesa i promeni način rada.

- Prenos podataka nakon zahteva od strane korisnika (*capture mode*), pri čemu je: poznat broj slika koje je potrebno preneti, opseg slike koji se prenosi (da li se prenosi cela slika, ili samo deo slke koji predstavlja deo od interesa, *ROI – Region of Interest*). Nakon ispunjenog zahteva sistem se vraća u svoj osnovni način rada - *preview*.

5. AURORA SPREŽNI SISTEM

Aurora predstavlja prilagodljiv protokol koji se koristi za prenos podataka od tačke do tačke (*point-to-point*) serijskom vezom. Aurora je "otvoren" protokol, besplatno rešenje, i može biti implementiran u bilo kojoj integrisanoj komponenti. Obezbeđuje transparentnu spregu fizičkom serijskom vezom, omogućavajući višim nivoima, kao što su Ehternet i TCP/IP, da lako koriste serijsku vezu visoke propusne moći. Na taj način dobijena je veća povezanost i prilagodljivost dok je u isto vreme očuvana postojeća infrastruktura softvera. Aurora je veoma efikasan protokol sa malim kašnjenjem koji koristi minimalnu moguću logiku. Ona nudi veoma bogat, visoko konfigurablilan skup karakterisitika. Može biti iskorišćen u bilo kojim aplikacijama koje koriste od tačke do tačke tip konekcije.

Preduslov za korištenje Aurora jezgra u FPGA komponentama proizvođača Xilinx je da iste poseduju više gigabitne primopredajnike (*MGT – Multi Gigabit Transceiver*) na pojedinim pinovima. Pomenuti diferencijalni primopredajnici omogućuju veoma visoke učestanosti prenosa podataka (do 6.5GHz) i time obezbeđuju rad određenih paralelnih sprega. Aurora jezgro je generisano pomoću programskog alata *Xilinx Core Generator*.

6. KONTROLA TAKTA

Slika 2 prikazuje i podsistem za upravljanje taktom. Osnovni zadatak podsistema je da obezbedi precizno definisane i stabilne signale takta i sinhronizovani signal reseta za sve module sistema. Na predjoj ploči se nalaze dva kristalna oscilatora učestanosti 48MHz i 78MHz. Modul senzor upravljača i DAC upravljača sinhronizovani su signalom takta učestanosti 66MHz. Pošto Aurora sprežni podsistem radi sa taktom učestanosti 78MHz neophodno je bilo obezbediti resinhronizaciju podataka i komandi pri prelazu između takt domena. Problem je rešen upotrebom memorijske strukture tipa asinhronog reda (*fifo first in - first out structure*). U okviru sistema realizovane su dve FIFO memorijske strukture: jedna magacinska memorija za smeštanje podataka od senzora i druga magacinska memorija za smeštanje komandi. Magacinska memorija podataka objedinjuje donjih 100 bita koji predstavljaju informacije sa senzora, kao i statusne informacije senzor upravljača gornjih 28 bita ka sprežnom sistemu (ukupno 128 bita). Druga magacinska memeorija prosleđuje senzor upravljaču komande od PC aplikacije preko Aurora spredng sistema.

8. ZAKLJUČAK

Realizovani sistem predstavlja jedno rešenje implementacije upravljača za CMOS senzor, sa priloženim skupom komandi kao i inicijalno zahtevanim karakterisitikama. Postojeću arhitekturu komandi moguće je proširiti novim zahtevima, kao i karakteristikama postojećih načina rada.

LITERATURA

[1] "MAPS High Speed Camera Design Specification" *RT-RK Computer Based Systems*, Novi sad, Maj 2007.

[2] "High Speed CMOS Image Sensor - Datasheet", *Micron*, 2007.

[3] "LogiCOREAurora v2.6 - User Guide", *Xilinx*, 2007.

[4] "DACs AD5307/AD5317/AD5327", *Analog Devices Inc.,* 2000.

[5] "Xilinx Core Generator", *Xilinx*, 2007.

Abstract – This paper describes one solution of a FPGA based system. The system represents high speed, high definition camera. A subsystem that controls and manages data transfer from commercialy available sensor and adaptes the data format for high-speed serial connection is described in detail.

ONE SOLUTION OF CMOS SENSOR CONTROLLER

Dragan Topalović, Dušan Majstorović, Zoltan Pele, Predrag Eremić

JEDNO REŠENJE IMPLEMENTACIJE KOMUNIKACIONOG KONTROLERA ZA *BBT2 SDK* ISPITNI SISTEM

Stevan Perišić, *Fakultet tehničkih nauka, Novi Sad,* stevan.perisic@micronasnit.com
Tomislav Maruna, *MicronasNIT, Novi Sad,* tomislav.maruna@micronas.com
Velibor Mihić, *MicronasNIT, Novi Sad,* velibor.mihic@micronas.com
Vukota Peković, *MicronasNIT, Novi Sad,* vukota.pekovic@micronas.com

Sadržaj *— Prikazano je jedno rešenje komunikacionog kontrolera za generisanje i kontrolu emitovanja digitalnog video signala iz BBT2 SDK sistema. BBT2 SDK je sistem za automatsko ispitivanje uređaja po principu crne kutije. U radu su opisani principi rada BBT2 sistema kao i detalji implementacije komunikacionog kontrolera za DekTec DTA-110T video karticu.*

1. UVOD

Da bi se što efikasnije ispitali uređaji i programske podrške, razvijeni za potrebe TV industrije, razvijen je *BBT2 SDK* sistem za ispitivanje. Ovaj sistem radi po principu crne kutije. Poznati su ulazni (stimulusi) kao i referentni izlazni podaci. Princip rada samog uređaja je nepoznat i nebitan za tok i rezultate ispitivanja. Stanja na izlazima ispitivanog uređaja, dobijena delovanjem stimulusa na njegovim ulazima, se snimaju. Upoređivanjem dobijenih sa referentnim izlaznim podacima se utvrđuje ispravnost rada ispitivanog uređaja.

BBT2 koristi uređaje različitog tipa za stimulaciju ulaza (između ostalog i *DekTec* video karticu). Takođe, koriste se i uređaji za snimanje stanja na izlazima ispitivanog uređaja. Stanja, otkrivena na izlazima ispitivanog uređaja, porede se sa očekivanim stanjima na izlazima. Na osnovu ovog poređenja donosi se zaključak o ispravnosti rada ispitivanog uređaja.

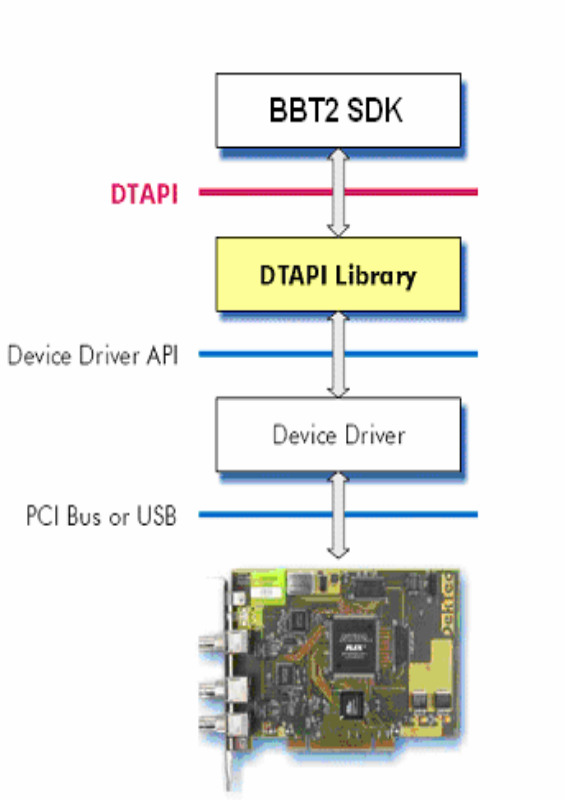
DekTec DTA-110T je digitalna video kartica koja radi kao DVB-T, DVB-H ili QAM A/B/C modulator sa UHF pretvaračem. Ona generiše modulisani RF signal u [400 MHz – 862 MHz] frekventnom opsegu. Za komunikaciju sa *BBT2* sistemom koristi PCI magistralu. Ova kartica i *BBT2* sistem su instalirani na istom računaru. Kontrola nad radom kartice se ostvaruje upotrebom odgovarajućih funkcija iz *DekTec* API-ja. *DekTec* API se oslanja na kontroler kartice.

DTA-110T komunikacioni kontroler je jedan od kontrolera koje *BBT2* koristi u svom radu. Zadatak ovog kontrolera je da kontroliše rad *DekTec DTA-110T* video kartice. Ova kartica se koristi kao ulazni uređaj u procesu ispitivanja. Korisniku *BBT2* sistema se mora obezbediti mogućnost da, upotrebom ovog kontrolera, kreira i dovodi digitalni video signal na ulaz ispitivanog uređaja. To može biti TV prijemnik ili neki drugi uređaj koji, kao jedan od ulaza, koristi video signal generisan po DVB standardu.

Oslanjajući se na funkcije *DekTec* API-ja, ovaj kontroler u potpunosti omogućava upotrebu svih resursa video kartice. Korisnik *BBT2* sistema koristi ovaj kontroler na standardan način, karakterističan za ovaj sistem.

Obezbeđen je dualni pristup funkcijama *DekTec* API-ja. Prvi omogućava direktnu kontrolu nad svim funkcijama API-ja. Drugi pristup pojednostavljuje rad sa karticom. To je omogućeno tako što su podešavanja kompleksnih parametara grupisana u nekoliko jednostavnih funkcija. Te funkcije su na raspolaganju korisniku sistema.

Na slici 1. je prikazan, već opisani, princip komunikacije između *BBT2 SDK* sistema za ispitivanje i *DekTec* digitalne video kartice.



Sl.1. Hijerarhija programskih slojeva rešenja

2. OPIS BBT2 SDK SISTEMA

BBT2 (**B**lack **B**ox **T**esting 2) sistem kontroliše uređaje iz ispitnog okruženja preko njihovih komunikacionih kontrolera. Svaki komunikacioni kontroler je napisan kao biblioteka sa dinamičkim uvezivanjem (DLL). Svrha ovakvog kontrolera je da omogući potpunu kontrolu nad radom uređaja za koji je napisan. Takođe, kontroler mora obezbediti komunikaciju sa *BBT2* sistemom putem uniformnog protokola.

BBT2 sistem, po svom pokretanju, dinamički uključuje sve prisutne komunikacione kontrolere. Ti kontroleri moraju biti na lokaciji određenoj za biblioteke komunikacionih kontrolera. Kontroler novog uređaja može biti priključen sistemu tako što se smesti na tu lokaciju. Jedna biblioteka

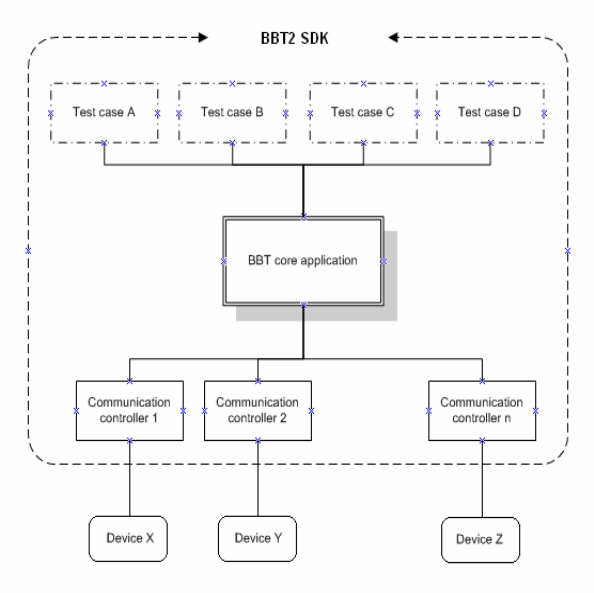
Napomene:
a) Rad je proistekao iz diplomskog-master rada Stevana Perišića. Mentor je bio prof.dr Nikola Teslić.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

može implementirati jedan kontroler (što je najčešći slučaj). Međutim, biblioteka može implementirati i više kontrolera, ukoliko postoji logička veza između njih.

Ovakav princip omogućava jednostavno podešavanje funkcionalnosti *BBT2* sistema. To je omogućeno dinamičkim uključivanjem, odnosno isključivanjem, kontrolera nekog od podržanih uređaja. Takođe, omogućeno je proširenje *BBT2* sistema širokim spektrom uređaja. To se ostvaruje pisanjem odgovarajućeg komunikacionog kontrolera za svaki od potrebnih uređaja.

Korisnik *BBT2* sistema kreira ispitne slučajeve iz kojih se upravlja radom potrebnih uređaja. To radi tako što u ispitnim koracima poziva funkcije odgovarajućih komunikacionih kontrolera. Po pozivu funkcije nekog komunikacionog kontrolera on preuzima prosleđenu promenljivu. U zavisnosti od vrednosti dobijene promenljive kontroler upravlja radom odgovarajućeg uređaja. Na taj način korisnik posredno upravlja radom ispitnih uređaja u procesu ispitivanja nekog uređaja.

Na slici 2. je dat grafički prikaz, gore opisane, organizacije *BBT2 SDK* sistema.



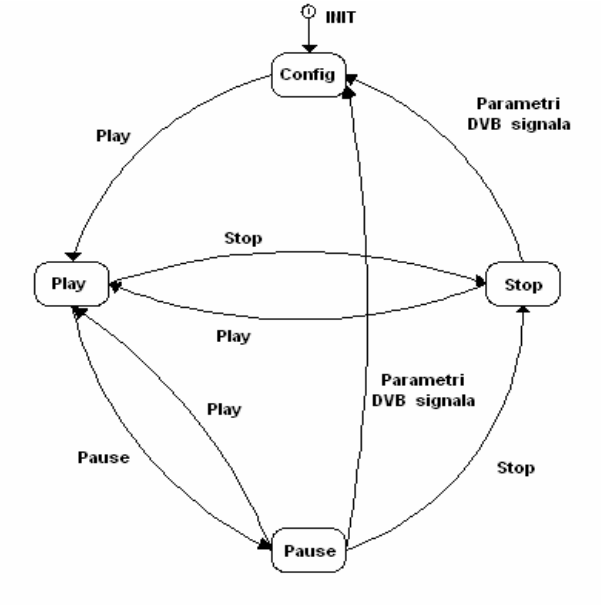
Sl.2. *Blok šema BBT2 SDK sistema*

3. OPIS REALIZACIJE KONTROLERA

Korisniku *BBT2* sistema omogućena su dva načina u radu sa ovim komunikacionim kontrolerom. Prvi, složeniji način, korisniku omogućava neposredan pristup svim funkcijama i parametrima *DekTec* API-ja. Ovaj pristup zahteva temeljno poznavanje svih resursa i rada same kartice. Pored toga, korisnik rukuje sa puno parametara u radu. Drugi, jednostavniji način rada sa komunikacionim kontrolerom omogućava upotrebu ove video kartice širem krugu korisnika. Upravo zato, u ovom radu će biti opisan drugi pristup.

Apstrakcijom funkcija video kartice kompleksna podešavanja parametara rada *DekTec* video kartice su sakrivena od korisnika. Korisniku je omogućeno podešavanje osnovnih parametara za generisanje digitalnog video signala (*Frequency*, *ModulationType*, *InputFile*…). Pored toga, omogućena mu je upotreba osnovnih komandi za emitovanje dobijenog video signala (*Play*, *Stop*, *Pause*).

Da bi upotreba kontrolera bila što jednostavnija (i robusnija) implementiran je automat stanja. Na slici 3. se vide stanja u kojim se automat može nalaziti. Prikazane su i promenljive čijim izmenama se prelazi iz jednog u drugo stanje automata.



Sl.3. *Automat stanja komunikacionog kontrolera*

Parametri DVB signala su:

- *InputFile* – putanja do datoteke sa video sadržajem u MPEG formatu
- *BitRate* – bitska brzina signala (u 0 – 31700000 bit/s opsegu)
- *Frequency* – frekvencija nosioca signala (u 400 MHz – 862 MHz opsegu)
- *TransmitMode* – tip paketa za slanje (188:0, 188:1, ADD16:0, ADD16:1, 204:0, 204:1, RAW:0 ili RAW:1)
- *ModulationType* – tip modulacije (DVB-T, QAM16, QAM32, QAM64, QAM128 ili QAM256)
- *ConvolutionalRate* – odnos konvolucije (1/2, 2/3, 3/4, 5/6 ili 7/8)
- *Bandwith* – širina frekventnog spektra (5MHz, 6MHz, 7MHz ili 8MHz)
- *Constellation* – tip konstelacije (QPSK, QAM16 ili QAM64)
- *GuardInterval* – interval zaštite (1/4, 1/8, 1/16 ili 1/32)

- *Interleaving* – umetanje signala (NATIVE ili INDEPTH)
- *TransmissionMode* – tip transmisije (2KB, 4KB ili 8KB)
- *DVB-H* – koristi se za preciziranje tipa signalizacije kod DVB-H standarda (ENA4849, DIS4849, S48_OFF, S48, S49_OFF ili S49)
- *Cell* – identifikacioni broj DVB-H ćelije (u 0 – 4294967295 opsegu)
- *J83Annex* – tip J.83 aneksa (A, B ili C)

Promena vrednosti ovih promenljivih nije moguća ukoliko se automat nalazi u *Play* stanju. Da bi došlo do promene vrednosti neke promenljive neophodno je preći iz *Play* u neko od naredna dva stanja. Do prelaska iz *Play* u neko drugo stanje se dolazi upotrebom promenljivih *Pause* ili *Stop*. Po prelasku automata u *Pause* ili *Stop* stanje, moguće je promeniti vrednost nekog parametra DVB signala. Ovim se obezbeđuje da do promene parametara video signala ne dolazi u toku emitovaja istog.

Promena nekog od nabrojanih parametara dovodi do prelaska automata stanja iz *Stop* ili *Pause* u *Config* stanje. U *Config* stanju je moguće promeniti vrednosti više različitih promenljivih.

Do emitovanja video signala se dolazi upotrebom promenljive *Start*.

Razlika između *Stop* i *Pause* stanja je u tome što se u *Pause* stanju pamti gde je emitovanje zaustavljeno. Sa emitovanjem se nastavlja od te tačke. U *Stop* stanju to nije slučaj, tj. emitovanje signala kreće ispočetka.

4. ISPITIVANJE I VERIFIKACIJA

Komunikacioni kontroler je ispitan kao deo *BBT2 SDK* isptnog sistema. U toku ispitivanja upotrebljena su i dva različita uređaja. Radi se o *Samsung* digitalnom TV prijemniku i *Cinergy 1400 DVB-T* demodulatorskoj PCI kartici. Ova kartica je instalirana na isti računar kao i *DekTec DTA-110T* modulatorska PCI kartica i *BBT2* sistem.

Navedeni uređaji su imali ulogu objekta ispitivanja od strane *BBT2* sistema. Jasno je da je njihova funkcija prikaz primljenog video signala. U slučaju ispitivanja sa TV prijemnikom signal je prikazan na njegovom ekranu. U slučaju ispitivanja sa *Cinergy* demodulatorskom karticom upotrebljena je programska podrška koja je prikazivala video sadržaj na ekranu monitora.

Kreirano je više ispitnih slučajeva pomoću kojih su isprobane situacije u kojima je najčešće moglo doći do greške u radu komunikacionog kontrolera. Neke od tih situacija su:

- pokretanje ispitnog sistema bez prisustva *DekTec DTA-110T* modulatorske kartice na računaru
- pokretanje ispitnog sistema u situaciji kada kontrolu nad resursima modulatorske kartice ima neka druga aplikacija
- promena vrednosti neke promenljive u toku emitovanja signala (automat u *Play* stanju)
- dodela neispravne vrednosti nekoj promenljivoj
- nepostojanje ulazne datoteke sa video sadržajem u MPEG formatu

Kontroler je u svim situacijama radio korektno.

5. ZAKLJUČAK

U ovom radu je objašnjen jedan pristup implementiranju komunikacionog kontrolera jednog ispitnog uređaja. Prikazano je kako ovakav kontroler prilagođava kontrolu nad resursima i funkcijama uređaja potrebama programskog sistema za koji je implementiran.

Takođe, opisan je način na koji funkcioniše ispitivanje po principu crne kutije. Opisom *BBT SDK* sistema, prikazana je jedna od mogućih implementacija ovakvog ispitnog sistema.

DTA-110T komunikacioni kontroler se oslanja na funkcije *DekTec* API-ja. Upotrebom ovih funkcija omogućena je upotreba svih resursa i funkcija *DekTec DTA-110T* modulatorske kartice. *DekTec* API omogućava kontrolu nad kompletnom porodicom *DekTec* video kartica. Neznatnim izmenama kontrolera omogućila bi se upotreba svih *DekTec* video kartica iz *BBT2* sistema.

Dodatno proširenje funkcionalnosti ovog kontrolera je moguće obradom ulaznih podataka potrebnih za generisanje video signala.

Ispitivanjem izolovanog kontrolera, kao i ispitivanjem u sklopu *BBT2* sistema, potvrđena je ispravnost rada *DTA-110T* komunikacionog kontrolera.

LITERATURA

[1] DekTec Digital Video BV, "DTAPI-C++ API for Dektec Devices"

[2] MicronasNIT, "DTA_110T DEVICE DRIVER [User Guide for Test Creation]"

[3] MicronasNIT, "BBT2 SDK Development - Requirements Specification "

[4] MicronasNIT, "BBT2 SDK Software Development Kit for Black Box Testing System [User Guide]"

[5] MicronasNIT, "NIT BB Testing Introduction "

[6] MicronasNIT, "NIT BB Testing Case Study "

Abstract – This paper gives one solution of communication controller implementation for *BBT2 SDK* testing system. This controller is used to control generation and emitting of a DVB signal. The paper gives a description of the *BBT2 SDK* system architecture and details of the controller implementation.

ONE SOLUTION OF COMMUNICATION CONTROLLER IMPLEMENTATION FOR *BBT2 SDK* TESTING SYSTEM

Stevan Perišić, Tomislav Maruna,

Velibor Mihić, Vukota Peković

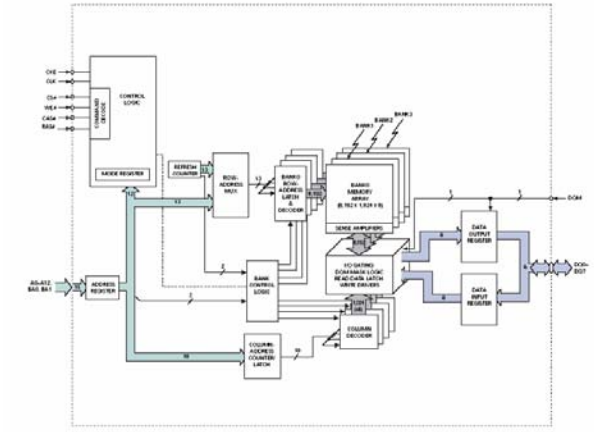
AUTOMATSKO ISPITIVANJE ISPRAVNOSTI UPISA I ČITANJA IZ SDRAM-A U *FULL PAGE BURST* MODU

Snežana Milosavljević, *Fakultet Tehničkih Nauka u Novom Sadu, Katedra za računarsku tehniku i računarske komunikacije,*
Miloš Nikolić, Zoltan Pele, *MicronasNIT Novi Sad*

Sadržaj – *U radu je predstavljeno jedno rešenje za automatsko ispitivanje ispravnosti upisa i čitanja iz SDRAM-a u full page burst modu. Na početku rada dat je kratak opis samog full page burst moda kao i način konfigurisanja memorije za rad u ovom režimu, dok je nastavak orijentisan na prikaz interne struktrure programabilne komponente FPGA.*

1. UVOD

Ispitivani memorijski modul je 256MB SDRAM, MT8LSDT3264AY, proizvođača Micron (detaljniji opis nalazi se u [1]), konfiguracije x64, organizovan u 4 banke, od kojih je svaka dalje organizovana u 8192 reda po 1024 kolone, tačnije po 1024 64-bitne lokacije (Sl. 1.).



Sl. 1. *32 Meg x 8 SDRAM blok dijagram*

Cilj ispitivanja je verifikacija funkcionalnosti navedenog SDRAM-a, upravljanim od strane SDRAM memorijskog kontrolera, kada je konfigurisan za rad u jednom od najređe korišćenih režima rada, tj. u *full page burst mode*-u. Za realizaciju ispitivanja iskorišćena je tehnologija programabilnih sekvencijalnih mreža FPGA (Field Programmable Gate Arrays), u ovom slučaju programabilna sekvencijalna mreža Spartan3 proizvođača Xilinx sa oznakom XC3S4000.

Ispitivanje obuhvata upis podataka (generisanih takođe u okviru ispitivanja) u kompletnu memoriju, zatim njihovo iščitavanje iz memorije i upoređivanje sa očekivanim izlazom, kao i generisanje signala greške ukoliko do nje dođe.

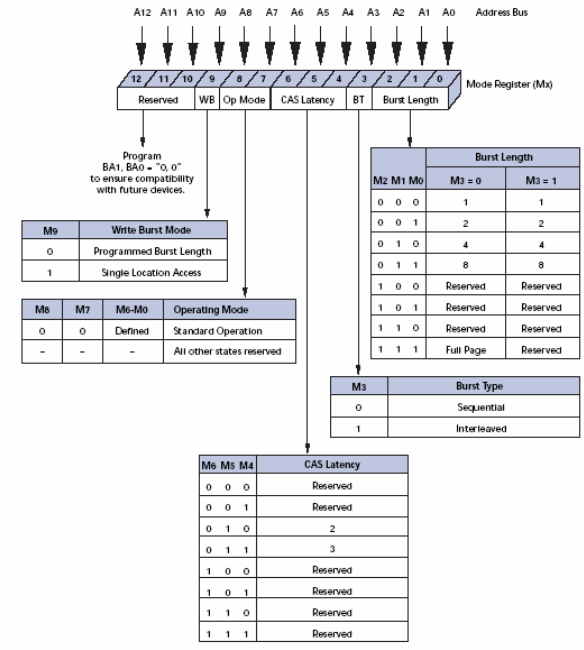
2. DEFINISANJE REŽIMA RADA SDRAM-A

Zadata komanda za upis ili čitanje iz memorije može se odnositi na pojedinačne memorijske lokacije, ali je češći slučaj da se jedna komanda odnosi na blok od 2, 4 ili 8 memorijskih lokacija (*burst mode)*. Za razliku od prethodno navedenih mogućnosti, ovde je obrađen slučaj kada se jedna zadata komanda, za upis ili čitanje, odnosi na ceo red (1024 uzastopne adrese) tzv. *full page burst mode*.

Odabir načina rada SDRAM-a, odnosno podešavanje broja memorijskih lokacija kojima se može pristupiti tokom izvršenja jedne operacije upisa ili čitanja, obavlja se programiranjem registra radnog režima (*Mode Register)* pre početka bilo koje druge operacije nad bankama SDRAM-a.

Registar radnog režima (Sl. 2.) je trinaestobitni registar čija 3 bita najniže vrednosti (*M2-M0*) određuju širinu bloka za pristup u toku jedne operacije pisanja ili čitanja (*burst)*. Sadržaj koji treba upisati u ovaj registar prosleđuje se adresnom magistralom istovremeno sa zadavanjem komande *LOAD MODE REGISTER,* što je detaljno opisano u [2].

Za iniciranje ove komande i postavljanje odredjenog podatka na adresnu magistralu zadužen je SDRAM kontroler. Stoga, za potrebe *full page* pisanja i čitanja bilo je potrebno izvršiti i određenu izmenu u samom kontroleru. Izmena se ogleda u tome da je, u automatu sa konačnim brojem stanja, koji kao izlaz generiše podatke koji se šalju na adresnu magistralu ka SDRAM-u, kada se on nalazi u stanju punjenja registra radnog režima, potrebno, 3 bita najmanje težine ovog izlaza postaviti na vrednost 1 (vidi sliku 2). Na slici 3 dat je segment koda koji predstavlja ovu izmenu.



Sl. 2. *Definicija mode registra*

Napomene:
a) Rad je proistekao iz diplomskog-master rada Snežane Milosavljević. Mentor je bio prof.dr Nikola Teslić.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

```
CASE state_d IS
  WHEN S_MODE_SET =>
    tmp_int_m_address (6 DOWNTO 4) := "010";  -- CAS latency time of 2 --full page
    tmp_int_m_address (2 DOWNTO 0) := "111";  -- CAS latency time of 2 --full page
```

Sl. 3. *Definisanje full page režima rada SDRAM-a*

3. *FULL PAGE* UPIS I ČITANJE

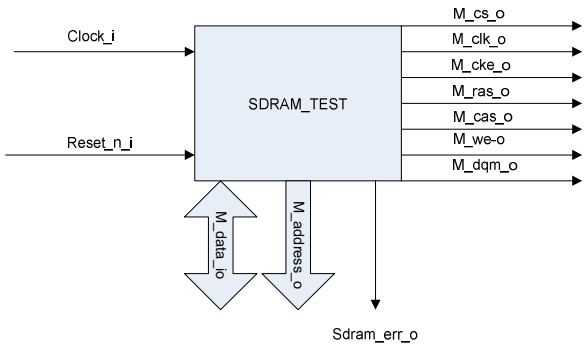
Iniciranje upisa i čitanja iz SDRAM-a, kao što je pokazano u [1] i [2], sastoji se u zadavanju komande *WRITE* (u slučaju pisanja u memoriju) odnosno komande *READ* (za čitanje iz memorije) i istovremenom postavljanju odgovarajuće adrese na adresnu magistralu. To zapravo znači, posmatrano sa višeg nivoa hijerarhije, da je na ulaz SDRAM kontrolera potrebno postaviti *sdc_req* na 1 (zahtev za pristupom memoriji), *sdc_rnw* na vrednost 1 u slučaju čitanja, odnosno 0 ukoliko se radi o upisu u memoriju, na *sdc_address* postaviti željenu adresu, i u slučaju pisanja, na ulaz *sdc_in_data* dovesti podatak za upis na zadatu adresu. Prilikom adresiranja treba imati u vidu da su adrese kontinualne, odnosno adresa poslednje kolone poslednjeg reda jedne banke je za 1 manja od početne adrese naredne banke tako da se prelaz između ovih banaka u navedenom slučaju može ostvariti prostim uvećanjem adrese *sdc_address* za 1. Gledano sa strane ulaza u samu SDRAM memoriju, biti adrese *m_address_o[12:0]* (izlaz iz SDRAM kontrolera, odnosno sprega ka SDRAM-u) koriste se za adresiranje reda i kolone jedne banke, dok biti *m_address_o[14:13]* adresiraju banku, [1] i [2].

Zadata operacija je uspešno obavljena samo ukoliko i *sdc_req* i *sdc_ack* (odgovor koji stiže od kontrolera) imaju vrednost 1. Kako je pristup memorijskim lokacijama SDRAM-a blokovski orijentisan *sdc_req* je potrebno držati na 1 da bi se pristup nastavio do kraja bloka ili do željenog dela bloka i tom prilikom uvećavati adresu sa svakim taktom (ukoliko je *sdc_ack* jednak 1) kao i dovoditi nove podatke za upis, na *sdc_in_data*.

Ukoliko je memorija konfigurisana tako da je moguće zaredom pristupati adresama u bloku od 2, 4 ili 8 lokacija, po završetku pristupa poslednjoj lokaciji u nizu, operacija se zaustavlja, tako da, npr. u slučaju pisanja u memoriju, podatak koji bi se pojavio na ulazu *sdc_in_data* po završetku pisanja na poslednju adresu bloka, bi bio ignorisan.

Ono što predstavlja razliku između pristupa memoriji u obliku *full page burst*-a, u odnosu na pristup jednoj lokaciji ili grupama od po 2, 4 ili 8, jeste karakteristika *full page* režima da se nakon izvršenja zadate operacije nad čitavim redom, pristup memoriji nastavlja cirkularno sve dok se ne zada komanda za prekid (*BURST TERMINATE)* ili se zatvori red kome se trenutno pristupa, [2], što se praktično može postići spuštanjem *sdc_req* na 0 na šta treba posebno obratiti pažnju radi pravovremenog obustavljanja operacije kako bi se izbeglo prepisivanje već popunjenih lokacija odnosno ponovno čitanje sa već iščitanih adresa.

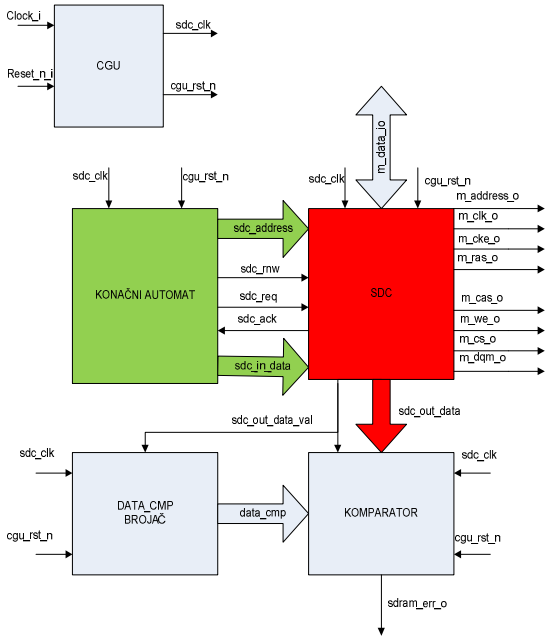
4. STRUKTURA PROGRAMABILNE KOMPONENTE FPGA



Sl. 4. *Vrh hijerarhije sdram ispitivanja*

Slika Sl. 4 prikazuje modul *sdram_test* koji predstavlja vrh hijerarhije i čiji su svi izlazi, izuzev *sdram_err_o,* sprega sa SDRAM-om. *Sdram_err_o* predstavlja signal greške. Ulazni takt (*clock_i)* je frekvencije 24 MHz.

Na slici Sl. 5 je data interna struktura programabilne komponente FPGA, proizvođača Xilinx sa oznakom XC3S4000.

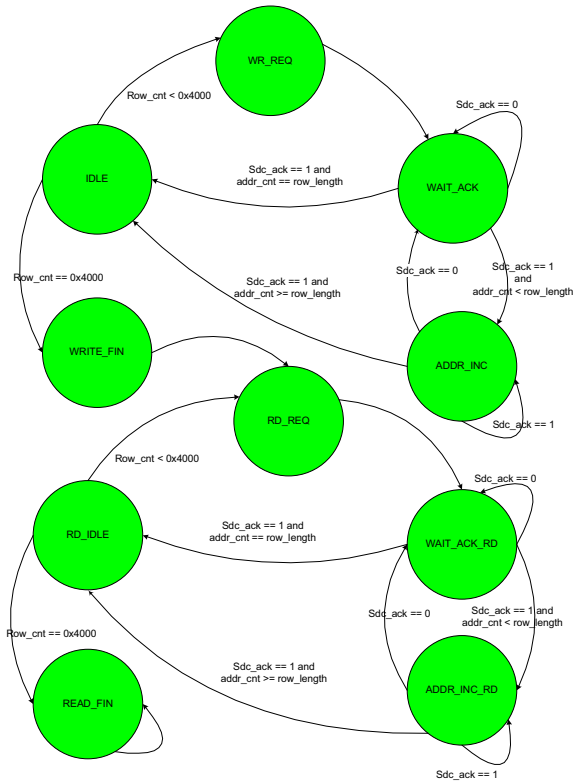


Sl. 5. *Interna struktura FPGA, Xilinx XC3S4000*

Struktura FPGA je realizovana pomoću Verilog jezika za opis fizičke arhitekture, izuzev SDRAM memorijskog kontrolera, koji je realizovan uz pomoć VHDL jezika.

Pored pomenutog modula (SDRAM kontrolera) internu strukturu FPGA čine i generator takta (*cgu*) koji sadrži standardni delitelj takta (DCM) kojim se ulazni takt frekvencije 24MHz pretvara u takt od 90MHz (*sdc_clk)* koji predstavlja ulazni takt za sve ostale module koji čine ovu strukturu. U strukturu, osim toga, ulaze i moduli brojača i komparatora (Sl. 5).

Izlaz iz jednog 64-bitnog brojača predstavlja referentni podatak sa kojim se poredi izlaz iz SDRAM memorijskog kontrolera (*sdc_out_data)*, odnosno podatak koji se pročita iz memorije. Ovakva realizacija referentnih izlaznih podataka je moguća, jer podaci koji se upisuju u memoriju predstavljaju takođe, uzastopne vrednosti brojača. Brojač generiše referentne podatke u toku čitanja, koristeći kao dozvolu brojanja signal *sdc_out_data_val,* koji je aktivan samo ukoliko je pročitani podatak sa zahtevane adrese, validan. U tom slučaju, vrednost brojača se uvećava za 1 i tako priprema odgovarajuća referentna vrednost za naredni podatak koji će biti pročitan. Brojač neće menjati izlaznu vrednost sve dok se ne pojavi novi validan podatak i obavi poređenje. Uloga komparatora je da uporedi ove dve vrednosti i kao izlaz generiše signal greške ukoliko se one razlikuju.



Sl. 6. *Dijagram prelaza stanja konačnog automata koji generiše ulaze u sdc*

Spregu ka SDRAM-u predstavljaju izlazi iz kontrolera, dok se ulazi u kontroler generišu u automatu sa konačnim brojem stanja. Dijagram prelaza stanja ovog automata dat je na slici Sl. 6. Automat se, na samom početku rada, kao i nakon reseta nalazi u stanju IDLE, pri čemu su svi izlazi (ulazi u SDRAM kontroler, kao i brojač kolona u jednom redu kojima se već pristupilo) postavljeni na nulu. Automat generiše i dozvolu brojanja za brojač redova kojima je već izvršen pristup. Kako se u stanje IDLE automat vraća nakon uspešno završenog pristupa jednom celom redu, tj. nakon završetka upisa u 1024 memorijske lokacije, vrednost ovog signala se postavlja na 1 i time se brojač upisanih redova uvećava za 1. Ukoliko je broj redova manji od 0x4000, što predstavlja ukupan broj redova memorije kojima treba pristupiti zbog upisa (ukupno 2 banke), a kasnije i čitanja, automat iz stanja IDLE prelazi u stanje WR_REQ, gde se zahtev za upisom u memoriju (*sdc_req)* postavlja na 1, a na izlaze koji predstavljaju adresu i podatke za memoriju postavljaju se početna adresa u redu kome treba pristupiti, i podaci koji se žele upisati.

Iz stanja WR_REQ prelazi se u stanje WAIT_ACK, u kome automat ostaje sve do pojave potvrde upisa od strane SDRAM kontrolera (*sdc_ack)* i pri tome ne menja vrednosti na izlazima. Signal potvrde (*sdc_ack)* ima ključnu ulogu u sinhronizaciji između podataka koje automat sa konačnim brojem stanja šalje na ulaz memorijskog kontrolera i podataka koji izlaze iz pomenutog kontrolera i postavljaju se na adresnu magistralu ka samoj memoriji. Naime, kontroler (*SDC*) postavlja signal *sdc_ack* na vrednost 1 samo ukoliko je podatak uspešno prosleđen ka memoriji i kontroler spreman za prihvatanje novog zahteva za pristupom SDRAM modulu. Nakon dobijanja potvrde od kontrolera, proverava se da li je brojač kolona dostigao vrednost maksimalnog broja kolona u jednom redu, tačnije 1024, i ukoliko jeste, automat prelazi u stanje IDLE (još jedan red upisan), inače, novo stanje je ADDR_INC. U ovom stanju uvećavaju se tekuća adresa i podatak za jedan. Zahtev za upisom se ne menja, odnosno zadržava vrednost 1 sve vreme i ukoliko potvrda od kontrolera (*sdc_ack)* stigne, automat ostaje u ovom stanju izuzev ako je brojač kolona dostigao vrednost 1024 kada se prelazi u stanje IDLE, a u slučaju kada nema potvrde, vraća se u stanje WAIT_ACK.

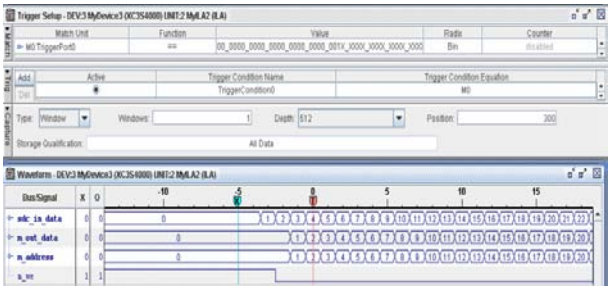
Ukoliko se u stanju IDLE ustanovi da je završen upis u celu memoriju, tj. da je brojač upisanih redova postao 0x4000, brojač redova se tada resetuje odnosno postavlja na nulu a automat prelazi u stanje WRITE_FIN u kome se svi izlazi takođe postavljaju na nulu. Treba napomenuti da je u svim do sada pomenutim stanjima signal *sdc_rnw,* koji određuje da li se upisuje ili čita iz memorije, bio postavljen na 0.

Iz stanja WRITE_FIN, prelazi se u stanje RD_REQ čime praktično započinje ciklus čitanja iz memorije. Uslovi prelaza stanja su isti kao i prilikom upisa : u stanju RD_REQ postavlja se zahtev za čitanjem (*sdc_req)* na 1, signal *sdc_rnw* takođe na 1 i zadaje se početna adresa u redu kome se pristupa (podaci za upis, *sdc_in_data,* nisu bitni jer se radi o čitanju); prelazi se u stanje WAIT_ACK_RD; ukoliko stigne potvrda uspešnog čitanja (*sdc_ack)* i brojač kolona je manji od 1024 prelazi se u stanje ADDR_INC_RD (ako je brojač jednak 1024 novo stanje je RD_IDLE); ako nema potvrde ostaje se u stanju WAIT_ACK_RD; tekuća adresa se uvećava u stanju ADDR_INC_RD iz koga se prelazi u stanje WAIT_ACK_RD ukoliko nema potvrde od kontrolera, a u stanje RD_IDLE ukoliko je završeno čitanje celog reda (brojač kolona jednak 1024); stanje RD_IDLE kod čitanja je ekvivalentno stanju IDLE kod pisanja; ukoliko se u njemu utvrdi da je brojač redova (pročitanih, u ovom slučaju) jednak 0x4000, automat prelazi u stanje READ_FIN u kome se i zadržava vraćajući sve izlaze ponovo na nulu.

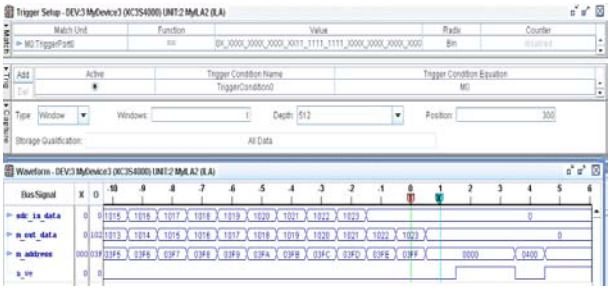
5. REZULTATI

Kako je ispitivanje realizovano tako da se podaci koji dolaze iz memorije, već u toku čitanja porede sa očekivanim izlazom, u slučaju neispravnog podatka, signal greške bi odmah bio postavljen na vrednost 1 ne menjajući je do kraja ispitivanja. Samim tim, uspešnost upisa i čitanja, bila bi verifikovana posmatranjem tog signala po završetku ispitivanja.

U ovom ispitivanju posmatranje rezultata je vršeno uz pomoć alata ChipScope Pro 9.2i, čije se uputsvo za upotrebu može naći u [3].



Sl. 7. *Upis u SDRAM-početak prvog reda*



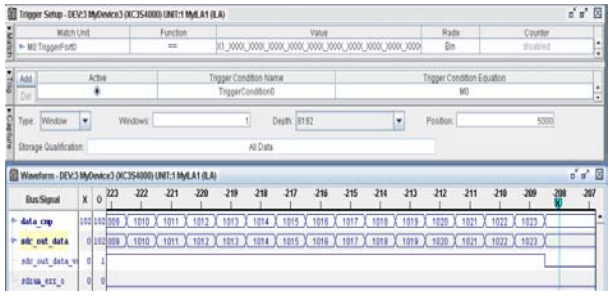
Sl. 8. *Upis u SDRAM-kraj prvog reda*

Tako, slika Sl. 12. predstavlja prozor gore pomenutog alata u kome se može očitati stanje signala greške po završetku ispitivanja, dok slike Sl. 7-11. predstavljaju neke od karakterističnih trenutaka u toku ispitivanja.

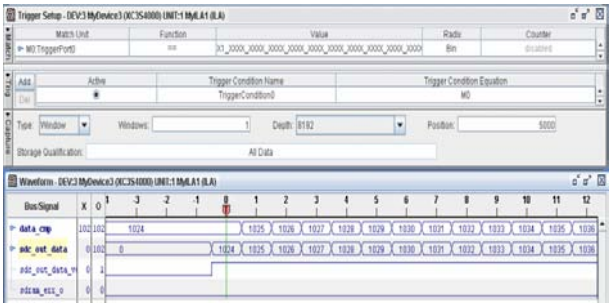
Na slikama Sl. 7 i Sl. 8 se mogu videti vrednosti ulaznih podataka kontrolera (*sdc_in_data*) za upis u memoriju i podataka koji idu ka SDRAM-u (*m_out_data*) kao i vrednosti adresa na koje će ovi podaci biti upisani (*m_address*).

Za razliku od njih, slike Sl. 9-11 prikazuju vrednosti pročitanih podataka (*sdc_out_data)* kao i očekivane vrednosti u tom trenutku *(data_cmp).* Podaci su validni dok je signal *sdc_out_data_val* (takođe prikazan na slikama Sl. 9-11) jednak 1.

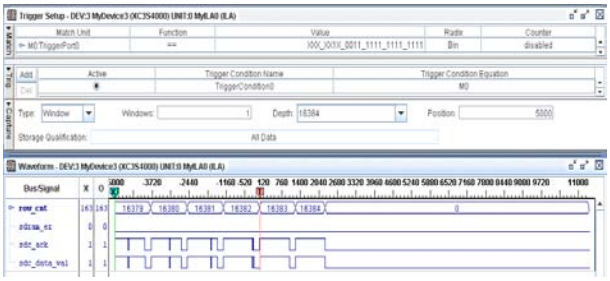
Sl. 9. *Čitanje prvog reda SDRAM-a*



Sl. 10. *Završetak čitanja prvog reda*



Sl. 11**.** *Početak čitanja narednog reda*



Sl. 12*. Signal greške na kraju testiranja*

6. ZAKLJUČAK

U okviru ovog rada prikazan je način automatskog ispitivanja upisa i čitanja iz SDRAM-a, MT8LSDT3264AY, proizvođača Micron, sa ciljem da se ispita ispravnost pomenutih funkcija memorije u full page režimu rada, ukoliko se pristup obavlja od strane programabilne sekvencijalne mreže Spartan3, proizvođača Xilinx, sa oznakom XC3S4000 na BBT (Black Box Testing Acqusition Board) platformi .

Kao što se može zapaziti na prikazanim slikama (tačnije Sl. 12), greška je jednaka nuli nakon završetka ispitivanja, tako da je ispravnost *full page* upisa i čitanja, u prethodno opisanim okolnostima, ovim ispitivanjem i potvrđena.

LITERATURA

[1] Micron: «Synchronous DRAM Module, MT8LSDT3264A(I)-256MB», www.micron.com/product/modules

[2] Micron: «Synchronous DRAM Module, MT48LC32M8A2–8Megx8x4banks», www.micron.com

[3] Xilinx: «ChipScope Pro Software and Cores User Guide», www.xilinx.com

Abstract – This paper describes one development of automatic verification of operations read and write to SDRAM in full page burst mode. It contains descriptions of full page burst, mode register programming, and internal structure of FPGA.

AUTOMATIC VERIFICATION OF OPERATIONS READ AND WRITE TO SDRAM IN FULL PAGE BURST MODE

Snežana Milosavljević, Miloš Nikolić, Zoltan Pele

PROGRAMABILNI EMULATOR DALJINSKOG UPRAVLJAČA

Darko Jambrek, Vukota Peković, Mihajlo Katona, Zoran Krajačević

***Sadržaj* — U ovom radu je izložena realizacija programabilnog emulatora uređaja za primanje i slanje komandi koristeći infracrvenu (IR) vezu (daljinskog upravljača). Opisan je način funkcionisanja uređaja, kao i programske podrške realizovane na mikrokontroleru i personalnom računaru potrebne za funkcionisanje ovog uređaja. Posebno je opisan proces primanja komandi, slanja komandi, kao i proces dodavanja novih komandi i protokola. Takođe su ukratko opisani trenutno dostupni komercijalni protokoli za slanje i prijem komandi korišćenjem IR veze.**

***Ključne reči* — IR – Infra Red (veza), Protokoli IR veze, UART – Universal Asynhronous Receiver/Transmitter.**

I. UVOD

Danas postoji veliki broj protokola koji se koriste za slanje podataka putem IR (Infra Red) veze. Većina renomiranih proizvodjaca kućnih aparata ima svoj protokol koji primenjuje u svojim uređajima. Zbog velike raznolikosti i varijacija u IR protokolima, a i iz potreba vezanih za kontrolu uređaja u procesu testiranja TV setova (uređaji koji ne poseduju konunikacionu spregu kojom bi se moglo upravljati uređajem, već se uređajem isključivo upravlja korišćenjem daljinskog upravljača) podstaklo je na realizaciju uređaja koji će pokušati u celini ili u većem delu da objedini IR protokole. Realizovani uređaj treba da obezbedi jednostavno dodavanje (obuku) novim protokolima, kao i mogućnost snimanja sekvence komandi poslatih sa drugog IR kontrolera. Emulator IR upravljača može da radi u dva osnovna režima:

- samostalan uređaj
- uređaj kojim se upravlja korišćenjem personalnog računara

Ovaj rad će se usredsrediti na režim rada uređaja kada je on u sprezi sa personalnim računarom.

Realizovani uređaj poseduje mikrokontroler koji je zadužen za dekodovanje primljenih IR komandi, kao i za kodovanje IR komandi u procesu slanja. Osim procesa prijema i slanja IR komandi, uređaj poseduje mogućnost dodavanja novih protokola.

Na tržištu postoje realizovani slični uređaji, ali je njihova primena ograničena. Postojeći uređaji šalju direktno sa IR senzora podatke na UART (Universal Asynchronous Receiver/Transmitter), a računar obavlja dekodovanje signala. Problem kod ovog pristupa je u tome što UART nije dovoljno brz da precizno prenese neobrađen signal, pa dolazi do grešaka pri dekodovanju. Ovakvi uređaji mogu se koristiti za daljinske upravljače koji se realizuju za potrebe upravljanja računarom, jer u takvim situacijama nije od velike važnosti da svaki poslata komanda bude obrađena. Ako slučajno dodje do greške pri obradi, tj. do neprihvatanja komande, komanda se može poslati ponovo. Međutim, ukoliko je od velike važnosti da svaka poslata komada bude adekvatno obrađena, tada korišćenje ovakvih uređaja nije podesno. Za potrebe testiranja drugih uređaja putem IR veze veoma je važno koje su komande primljene i kojim redosledom.

Iz tog razloga uređaj poseduje mikrokontroler koji dekoduje komande prilikom prijema, odnosno koduje komande prilikom slanja.

II. PROTOKOLI ZA IR KOMUNIKACIJU

Od protokola [2], [3] uređaj podržava RC-5 i RC-6, Sony SIRC, R-Step, IIT, JVC, NEC, Sharp, X-Sat, RECS80, Panasonic, Daewoo, RCA, RC-MM i Nokia NRC17. Nabrojani protokoli su međusobno različiti po nosećoj frekvenciji (36, 38, 40 i 56 kHz), po izgledu i trajanju signala nule i jedinice i po dužini znaka u bitima.

Po izgledu nule i jedinice protokoli se mogu svrstati u 3 grupe:

- prva grupa su protokoli kojima se nula i jedinica razlikuju u trajanju, tj. postoji impuls konstantne dužine nakon čega sledi pauza promenljive dužine
- Druga grupa su protokoli čije je trajanje i nule i jedinice isto, a razlika je u tome da li u sekvenci prvo ide pauza pa impuls ili impuls pa pauza.
- Treća grupa su protokoli koji za jednu kombinaciju impulsa i pauze salju po dva bita. Kod njih je, slično kao i kod prve grupe, impuls konstantne dužine, dok dužina pauze varira. U trećoj grupi protokola se nalazi samo RC-MM protokol.

III. PODRŽANI SCENARIJI

Dijagrami mogućih scenarija rada uređaja predstavljaju način rada uređaja i aplikacije na računaru, kao i njihovu međusobnu komunikaciju. Na sl. 1 prikazan je način na koji funkcionišu uređaj i aplikacija prilikom slanja i primanja komandi, kao i obučavanje. Obučavanje se koristi za dodavanje novih protokola i komandi. Razlikuje se deo koji se realizuje na računaru i deo koji se realizuje u mikrokontroleru uređaja.

Rad je delimično podržan u okviru projekta TR-6136B Ministarstva za nauku i zaštitu životne sredine Republike Srbije.

Darko Jambrek, Fakultet tehničkih nauka u Novom Sadu, odsek Elektrotehnika, Trg Dositeja Obradovića 6, Novi Sad, Telefon: 381-64-2465654; e-mail: darko.jambrek@gmail.com).

Vukota Pekovic, MicronasNIT, Fruskogorska 11, Novi Sad, Telefon: 381-21-4801118; e-mail: vukota.pekovic@micronas.com).

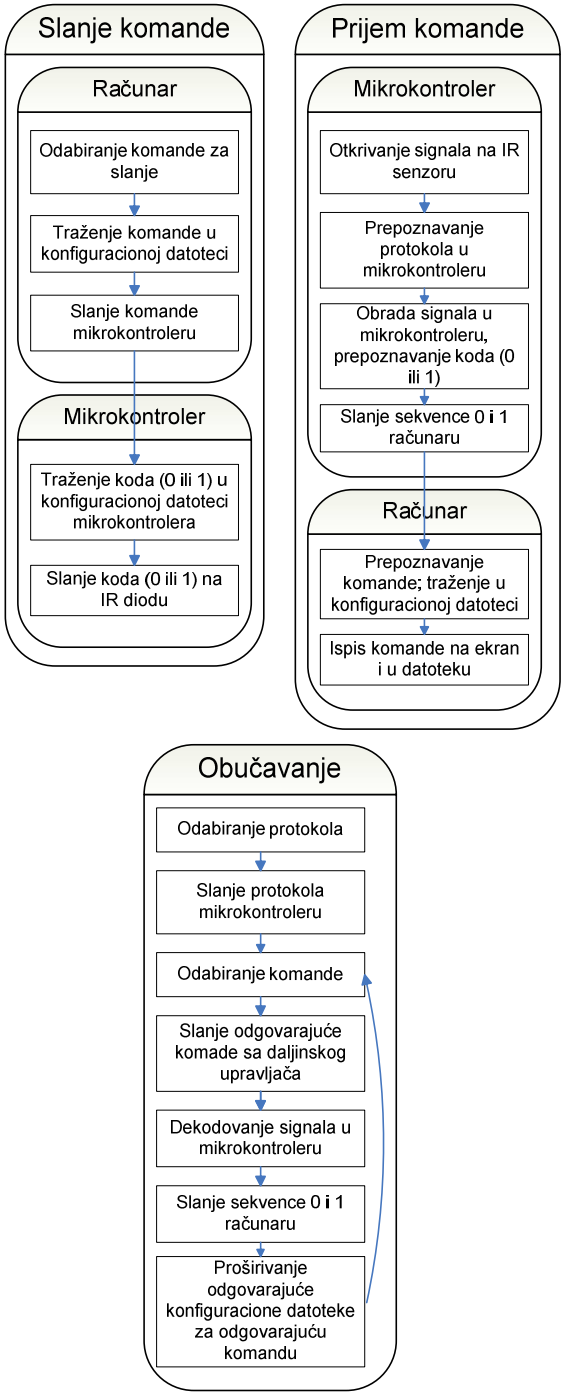
Mihajlo Katona, MicronasNIT, Fruskogorska 11, Novi Sad, Telefon: 381-21-4801178; e-mail: mihajlo.katona@micronasnit.com).

Zoran Krajačević, MicronasNIT, Fruskogorska 11, Novi Sad, Telefon: 381-21-4801181; e-mail: zoran.krajacevic@micronasnit.com).

Napomene:

a) Rad je proistekao iz diplomskog-master rada Darka Jambreka. Mentor je bio prof.dr Nikola Teslić.

b) Rad je prethodno publikovan na konferenciji TELFOR, Beograd, Novembar 2007.



Sl. 1. Dijagrami mogućih scenarija rada uređaja. Slanje, prijem komandi i obučavanje

IV. REALIZACIJA

A. *Mogućnosti uređaja*

Prva opcija koju uređaj treba da omogući jeste primanje i dekodovanje signala. IR komanda, poslata sa nekog daljinskog upravljača ili nekog drugog uređaja, se prima pomoću IR senzora. Osetljivost senzora mora da bude dovoljna da zadovolji spektar frekvencija koje zahtevaju različiti protokoli, a to je od 36 kHz do 56 kHz. U uređaju je korišten IR senzor od 38 kHz i radi korektno sa potrebnim frekvencijama.

Druga opcija uređaja je slanje komandi sa računara. Komande se šalju pomoću aplikacije koja se instalira na računar. Aplikacija odabrane komande prosleđuje uređaju.

Treća opcija je mogućnost obučavanja uređaja i aplikacije na računaru. Obučavanjem se dodaju novi protokoli i komande.

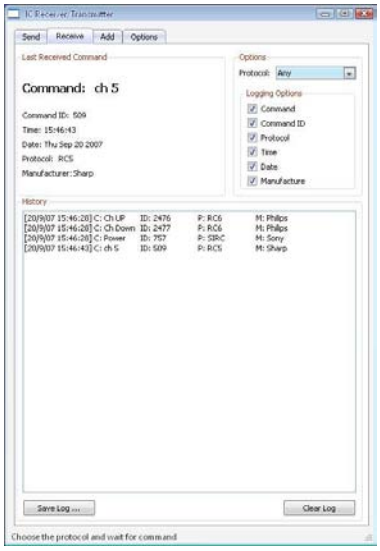
B. *Prijem signala*

IR senzor nakon otkrivanja IR komande pretvara svetlosne impulse u strujne i šalje ih ka mikrokontroleru [3]. Mikrokontroler od IR senzora prima niz impulsa određene frekvencije. Nakon toga sledi filtriranje po frekvenciji i poređenje dobijene frekvencije sa očekivanom. Očekivane frekvencije su frekvencije koje koriste IR protokoli, dakle, 36, 38, 40 i 56 kHz. Ukoliko se otkrije željena frekvencija sledi dalje prepoznavanje protokola, tj. trajanje impulsa i način njihovog pojavljivanja. Na osnovu ovih podataka može se proverom u tabeli protokola zaključiti koji protokol se koristi i pri slanju komande računaru poslati i aktuelni protokol. Kada se prepozna protokol sledi dekodovanje signala po normama odgovarajućeg protokola. Dekodovanje signala se sastoji u prepoznavanju nula i jedinica. Tako dekodovane nule i jedinice se šalju računaru na dalju obradu.

Podaci se od mikrokontrolera do računara šalju preko serijskog prolaza (port). Kada računar primi niz nula i jedinica od mikrokontrolera, sledi dalja obrada na računaru. Aplikacija realizovana na personalnom računaru proverava serijski prolaz i prima podatke sa njega. Mikrokontroler računaru šalje informaciju o kom se protokolu radi da bi PC (Personal Computer) aplikacija znala kako da informaciju dalje obradi. Kada PC aplikacija primi čitavu sekvencu nula i jedinica (ako se zna koji se protokol koristi, zna se i veličina komande u bitima) sledi prepoznavanje komande. Komanda se prepoznaje tako što aplikacija poredi primljenu komandu sa komandama koje se nalaze u unapred definisanoj tabeli. Kada se utvrdi o kojoj komandi se radi sledi njen ispis na ekran. Primer primljenih komandi ispisanih na ekran prikazan je na sl. 2. Aplikacija prikazuje poslednju primljenu komandu sa osnovnim informacijama (naziv komande, broj komande, naziv protokola, naziv proizvođača, datum i vreme prijema), kao i ostale primljene komande sortirane po vremenu prijema. Primljene komande se mogu sačuvati u posebnoj datoteci radi sigurnosti.

C. *Slanje signala*

PC aplikacija takođe omogućava da se pomoću nje šalju komande na uređaj, a uređaj pravi impulse IR svetlosti. Proces slanja komande je obrnut od procesa primanja komandi. Prvo sledi odabir protokola po kom se želi kodovati komanda. Nakon odabira protokola aplikacija šalje informaciju mikrokontroleru o kom protokolu je reč.



Sl. 2. Aplikacija na računaru, deo za prijem komandi

Sl. 3. Aplikacija na računaru, deo za slanje komandi

Komunikacija između aplikacije na računaru i mikrokontrolera na uređaju se takođe omogućuje putem serijskog prolaza. Nakon odabira protokola sledi izbor komande koja želi da se pošalje. Kada se odabere komanda, aplikacija traži odabranu komandu u tabeli komandi i utvrđuje odgovarajuću sekvencu nula i jedinica. Potom se sekvenca nula i jedinica šalje mikrokontroleru.

Nakon što mikrokontroler primi informaciju po kom protokolu treba da koduje primljene komande, on čeka sekvencu nula i jedinica koje salje računar. Primljenu sekvencu mikrokontroler koduje u realnom vremenu. Kodovanje sekvence nula i jedinica se sastoji u tome što mikrokontroler proveri tabelu protokola i ustanovi normu za kodovanje nule i jedinice. U trenutku kada mikrokontroler primi nulu ili jedinicu on je koduje po odgovarajućoj normi, tj. kreira diskretizovan analogni signal koji se sastoji od impulsa odgovarajuće frekvencije. Takav diskretizovani analogni signal impulsa se šalje na IR diodu. IR dioda pretvara strujne impulse u svetlosne i takve svetlosne impulse mogu da primaju razni uređaji osetljivi na IR svetlost odgovarajućeg protokola.

Izgled aplikacije na računaru u delu za slanje komandi prikazan je na sl. 3.

D. Komunikacija uređaj - računar

Računar i uređaj komuniciraju preko serijskog prolaza. Prenos podataka između računara i uređaja je dvosmeran, što znači da mikrokontroler šalje podatke RS-232 (Recommended Standard 232) [4] čipu koji te podatke pretvara po serijskom protokolu i šalje ih ka računaru, a takođe RS-232 podatke poslate sa računara prosleđuje dalje mikrokontroleru. Pošto postoji dvosmerna komunikacija obezbeđen je mehanizam zaštite usled slanja i primanja komandi u isto vreme. Korisnik u aplikaciji ima mogućnost izbora prijema ili slanja komandi. Kada se odabere prijem, odnosno slanje, komande, aplikacija obaveštava mikrokontroler. Ukoliko je odabrano slanje komandi, uređaj postaje neosetljiv na dolazne komande.

E. Proširivost

Pošto uređaj poseduje interni EEPROM (Electrically Erasable Programmable Read-Only Memory) to omogućava veliku fleksibilnost uređaja kao i mogućnost menjanja koda i dodavanja podrške za nove protokole.

Uređaj i aplikacija poseduju mogućnost proširivanja tabela protokola i komandi. Proširivanje tabela komandi se naziva obučavanje aplikacije, a proširivanje broja protokola se naziva obučavanje uređaja. Kada se proširi broj protokola na uređaju, automatski se povećava broj protokola i u aplikaciji. Uređaj ne poseduje tabelu komandi tako da ne važi obrnut slučaj, da se prilikom povećavanja broja komandi u aplikaciji povećava i na uređaju. Ovim se omogućuje dodavanje novih protokola i proizvođača, kao i komandi.

Prilikom dodavanja komandi prvo se odabere protokol kome pripada komanda. Nakon odabiranja protokola on se salje mikrokontroleru, radi sinhronizacije mikrokontro-lera sa aplikacijom na računaru. Nakon toga se bira komanda, ako postoji u listi komandi, a ako ne, onda se kreira novi naziv komande. Nakon odabiranja očekivane komande ona se šalje sa daljinskog upravljača. Uređaj koji je u stanju obučavanja prima komandu sa daljinskog upravljača i na osnovu protokola koji mu je zadat od strane PC aplikacije otkriva primljenu komandu (signal) i šalje je PC aplikaciji. Dakle, kada se dodaje nova komanda mikrokontroler se ponaša kao da je u modu za prijem komande. Nakon dekodovanja signala i kreiranja niza nula i jedinica (kod komande) sledi slanje niza računaru. Obučavanje aplikacije sastoji se u proširivanju tabele komandi. Postoji i mogućnost ručnog proširivanja komandi menjanjem inicijalne datoteke u kojoj se nalaze sve komande. Datoteka se menja po određenim normama.

Dodavanje protokola se takođe započinje u računaru. Prvo se mikrokontroleru posalje naziv protokola, a zatim izgled signala nule i jedinice. Oba signala (0 i 1) se sastoje od impulsa i pauze određenog trajanja. Kod raznih protokola trajanje impulsa i pauze je različito, tako da se

mora nameštati posebno trajanje impulsa, a posebno trajanje pauze. Tako da se izgled nule ili jedinice sastoji u tome da se navede da li prvo nailazi impuls ili pauza i da se definiše trajanje impulsa i pauze. Nakon toga se definiše trajanje komande u bitima, da bi mikrokontroler znao kada je dekodovao čitavu komandu i da bi znao kad komandu treba da šalje aplikaciji. Nakon podešavanja svih stavki protokola, te stavke se šalju mikrokontroleru koji prošuruje svoju tabelu protokola za novi protokol. U tabeli protokola u mikrokontroleru se nalaze za svaki protokol identične stavke koje su dodate. Ovim se završava proces dodavanja protokola.

F. Vremenski brojači (Timer-i)

Mikrokontroler [3] poseduje precizne vremenske brojače posto je vreme impulseva signala izraženo u mikro-sekundama, a od presudnog značaja je da se precizno izmeri trajanje impulsa. Kod mnogih protokola je razlika između nule i jedinice baš u trajanju impulsa ili pauze nakon impulsa. Svaki put nakon promene između impulsa i pauze startuje se odgovarajući vremenski brojač i nakon završetka impulsa ili pauze se zaustavi. Na taj način se meri vreme trajanja impulsa ili pauze. Ovo vreme trajanja impulseva i pauza pomaže u prepoznavanju IR protokola.

V. PRAKTIČNA PRIMENA

Zbog velike rasprostranjenosti kućnih aparata koji se kontrolišu putem daljinskog upravljača ovaj uređaj pronalazi svoje mesto u mnogim sferama tehničke primene.

Mogućnosti ovog uređaja su višestruke:

- zamena više postojećih daljinskih upravljača univerzalnim rešenjem
- potrebe testiranja drugih uređaja (koji koriste IR vezu za komunikaciju), kao i upravljanje tim uređajima
- upravljanje računarom pomoću daljinskog upravljača

Sa opcijom prijema IR signala ovaj uređaj se može koristiti za potrebe testiranja. Mogu se testirati kako daljinski upravljači tako i TV prijemnici. Opcija buffer-ovanja komandi je korisna iz razloga što kad se desi neko neočekivano stanje na tv-u može da se pogleda koje su sve komande korišćene i tako lakše utvrdi šta je prouzrokovalo to neočekivano stanje.

Sa opcijom slanja komandi uređaj se takođe može koristiti za potrebe testiranja. Uređaj može da služi kao univerzalni daljinski upravljač. Primena uređaja se proširuje tim što je uređaj povezan na računar, a računar se može isprogramirati da u određeno vreme šalje odgovarajuće komande, tako da se može koristiti i za automatizovano testiranje.

VI. ZAKLJUČAK

Ovaj uređaj je pogodan za korištenje, jer ima mogućnost slanja komadi, primanja komandi kao i mogućnost dodatnog obučavanja uređja. Većina trenutno dostupnih uređaja podržava samo jednosmernu komunikaciju. Navedeni uređaj je pogodan za testiranje širokog spektra tehničkih uređaja koji kao kontrolnu spregu (interface) koriste IR komunikaciju. Mogućnost proširivanja protokola i komandi mu dozvoljava široku primenu u automatskom testiranju televizora, DVD (Digital Versatile Disk) uređaja, STB (Set-Top Box), muzičkih linija i ostalih uređaja koji koriste daljinske upravljače.

LITERATURA

[1] http://www.sbprojects.com/knowledge/ir/ir.htm SB-Projects: IR remote control

[2] http://users.pandora.be/davshomepage/ Davshomepage

[3] http://secure.robotstore.com/download/248073.pdf PIC18FXX2 data sheet

[4] http://www.maxim-ic.com/appnotes.cfm/appnote_number/723/ Selecting and Using RS-232, RS-422, and RS-485 Serial Data Standards

ABSTRACT

This paper deals with the realization of programmable device emulator for commands sending and receiving via infra red connection (remote controller). In the paper there is described the emulator's functionality and software for both microcontroller and PC side of the application. The focus is on the command receiving and sending processes and the way for adding new commands and protocols. The text contains short description of currently available IR protocols, too.

PROGRAMMABLE EMULATOR OF REMOTE CONTROLLER

Darko Jambrek, Vukota Peković, Mihajlo Katona, Zoran Krajačević

AUTENTIFIKACIJA I AUTORIZACIJA U DISTRIBUIRANIM SISTEMIMA NA PRIMERU DCOM-A I CORBA-E

Uglješa Novak, *Fakultet tehničkih nauka, novakovi@neobee.net*
Srđan Orlović, *DMS Group*, *srdjan.orlovic@dmsgroup.co.yu*
Nikola Radovanović, *DMS Group*, *nikola.radovanovic@dmsgroup.co.yu*

Sadržaj – *U ovom radu su predstavljeni osnovni principi sigurnosti, tj. autentifikacije i autorizacije, u distribuiranim sistemima i njihova realizacija u DCOM-u i CORBA-i. Cilj ovakvog prikaza jeste da se opiše postupak autentifikacije i autorizacije kroz preuzimanje i delegiranje klijentskih ovlašćenja (credentials), kao i da se ispita da li CORBA podsistem zaštite omogućava ili ne sve funkcionalnosti koje su ostvarene pomoću/upotrebom DCOM-a (odnosno da se izvrši uporedna analiza).*

1. UVOD

Ovo istraživanje je inicirano potrebom migracije *software*-skih komponenti *DMS Group* programskog paketa sa DCOM (specifičnog za *Microsoft Windows* operativni sistem) standarda za klijent-server komunikaciju u distribuiranim sistemima na *cross-platform*, *open source* rešenja, pre svega CORBA standard.

1.1 AUTENTIFIKACIJA I AUTORIZACIJA

Uvođenje klijent-server arhitekture unelo je revoluciju u svetu računara i računarskih komunikacija, ali pored svih prednosti takođe su stvoreni i mnogi novi problemi. Jedan od tih problema svakako je i sigurnost. Distribuirani sistemi, zbog svoje prirode, su najpodložniji ovakvim problemima. Kao preduslov za uspostavljanje sigurnosnog distribuiranog sistema predstavlja implementacija autentifikacije i autorizacije lokalnih i/ili udaljenih entiteta u sistemu:

- Autentifikacija (*authentication*) u opštem smislu predstavlja identifikaciju i verifikaciju. Principal (čovek-korisnik ili drugi računar) koji zahteva neku uslugu (*service*), postavljanjem svog identiteta treba da dokaže da on zaista jeste validan korisnik tražene usluge. U opštem slučaju autentifikacija klijenta je zadatak autentifikacione usluge. Npr. NTLMSSP (*NT LAN Manager Security Support Provider*) koji predstavlja *Windows* standard.
- Autorizacija (*authorization*) se odnosi na (ne)odobravanje određenih usluga principalu koje je tražio, na osnovu rezultata autentifikacije, zahtevanih usluga i trenutnog stanja sistema. Autorizacija može biti zasnovana na različitim vrstama ograničenja (lokacija korisnika, trenutno vreme, zabrana višestrukog logovanja jednog te istog korisnika i sl.). Autorizacijom se određuje priroda usluge koja se daje na korišćenje.

2. DCOM

DCOM (*Distributed Component Object Model*) je *Microsoft*-ova tehnologija koja omogućava komunikaciju *software*-skih komponenti distribuiranih u zajedničkoj mreži. To je u osnovi mrežni protocol, visokog nivoa, koji omogućava komponentama zasnovanim na COM-u komunikaciju između procesa na različitim računarima.

2.1 OPŠTI SIGURNOSNI MEHANIZMI

DCOM pruža više nivoa zaštite. Mnogi mehanizmi zaštite u DCOM-u su preuzeti iz drugih podsistema, pre svega od RPC-a (*Remote Procedure Call*).

Jedna od osnovnih karakteristika koju koristi DCOM je da se korisniku prilikom logovanja na računar, tj. operativni sistem, pridružuje *access-token* koji se kasnije koristi tokom autentifikacije na serveru kao i pri proveri prava pristupa određenim resursima. Ovim resursima su pridruženi *security descriptor*-i odn. strukture koje sadrže informacije o pravima pristupa.

DCOM takođe nudi nekoliko mogućnosti korisniku, tj. programeru, pri izboru mehanizma zaštite:

- globalno: parametri zaštite se konfigurišu spolja, izvan samih komponenti, administrativnim alatima (*dcomcnfg*). Same aplikacije, tj. komponente, nisu svesne sigurnosnih podešavanja. Prednost ovog metoda je da se iste *software*-ske komponente mogu različito konfigurisati u različitim okruženjima. Ove konfiguracije se u DCOM-u primenjuju automatski i tako omogućuju veću fleksibilnost pri instalaciji (*deployment*) aplikacije.
- programski: parametri zaštite se definišu programski u samim komponentama, tj. u izvornom kodu. Podešavanje cele sigurnosne infrastrukture od strane programera prepušta klijentima i objektima da preuzmu programsku kontrolu nad njihovim sigurnosnim merama, tj. ograničenjima.

Opšti sigurnosni mehanizmi iniciraju se metodom *CoInitializeSecurity()*. Njom su omogućena pred-procesna podešavanja sigurnosnih nivoa, tj. omogućeno je autoru da specificira alternative za autentifikacionu uslugu (NTLMSSP, *Kerberos*, *Snego*, *Schannel*, itd.), autentifikacioni nivo i nivo delegacije klijentskih ovlašćenja. U slučaju da ova metoda nije pozvana, COM će procesu automatski postaviti sistemska, unapred definisana, sigurnosna podešavanja koja su zapisana u ključevima *Windows registry*-ja (*HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Ole*).

2.2 AUTENTIFIKACIJA I AUTORIZACIJA

Kad god klijentska aplikacija pozove metodu ili kreira instancu komponente servera, DCOM preuzima od *Windows Security*-a klijentska ovlašćenja i pridružuje ih datom procesu u vidu oznake (*token*). Za autentičnost kredincijala na odredištu takođe je odgovoran *Windows Security* kojem DCOM na strani servera prosleđuje oznaku. U slučaju prihvatanja ovlašćenja tj. potvrđivanja permisija klijenta, uspostavlja se komunikacija i odobrava uslugu. Takođe metodom *CoQueryClientBlanket()*, COM interfejs *IServerSecurity*, procesu poslužioca je omogućeno dobijanje informacije o ovlašćenjima klijenta. Da bi se izbegla stalno

---

Napomene:
a) Rad je proistekao iz diplomskog-master rada Uglješe Novaka. Mentor je bio prof.dr Branislav Atlagić.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

prosleđivanje oznaka, u DCOM-u je omogućeno navesti nivo na kome će se izvoditi autentifikacija korisnika, tj. provera ovlašćenja. Npr. autentifikacija klijenta može biti ograničena samo pri uspostavi veze dva računara gde server pamti oznaku klijenta i automatski ga koristi kada god otkrije zahtev od istog korisnika (*RPC_C_AUTHN_LEVEL_CONNECT*). Rukovanje klijentskim ovlašćenjima od strane servera takođe je omogućeno kroz COM interfejs *IServerSecurity*. Pored dobijanja informacija o ovlašćenjima ovom spregom server može da se „predstavi“ kao klijent preuzimanjem njegovih ovlašćenja, tj. oznake (metoda *CoImpersonateClient()*).

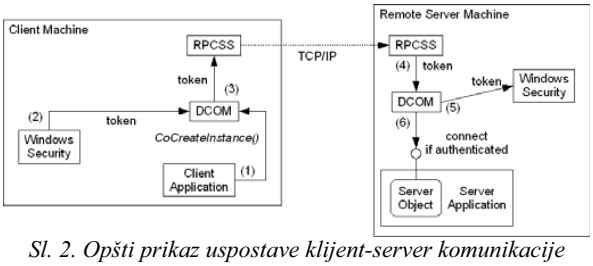
2.3 PRIMER

Za potrebe predstavljanja implementacije autentifikacije i autorizacije u DCOM-u razvijena je serverska aplikacija (COM *server*) koja omogućava ili ne pristup datoteci na udaljenom računaru, odn. udaljenom resursu, u zavisnosti od prava koja su konfigurisana za tu datoteku i korisnikovih ovlašćenja. Isto tako aplikacija pruža i mehanizam za preuzimanje korisnikovih ovlašćenja radi vođenja evidencije o pristupu datoteci (koji korisnik je i koliko pristupao datoteci). Takođe je razvijena i klijentska aplikacija za pristup serveru koja je pokretana na različitim računarima u mreži, kao i pod različitim korisničkim nalozima koji su definisani u domenu.

Parametari (nivoi) zaštite u klijentskom i kodu poslužioca su zadani pozivom *CoInitializeSecurity()* metode (Sl. 1). Klijent pozivom *CoCreateInstance()* metode počinje uspostavljanje sigurnosne komunikacije na zadatom nivou zaštite (Sl. 2).

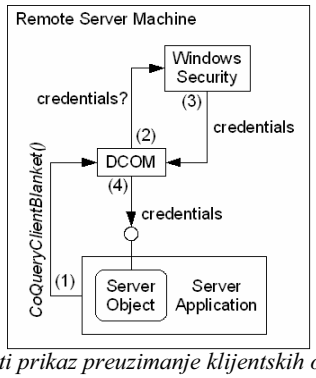
```
hr = ::CoInitializeSecurity(
            // Allows access to all users
            NULL,
            -1, NULL, NULL,
            // Client authentication only on connection
            RPC_C_AUTHN_LEVEL_CONNECT,
            // Allows server to impersonate client
            RPC_C_IMP_LEVEL_IMPERSONATE,
            NULL,
            EOAC_NONE,
            NULL
        );
```

Sl.1. Primer sigurnosnog podešavanja



Sl. 2. Opšti prikaz uspostave klijent-server komunikacije

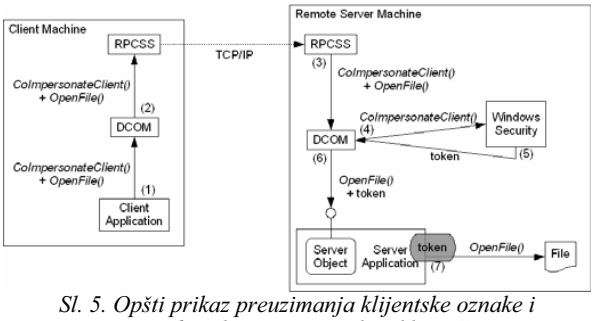
Preuzimanje korisničkih ovlašćenja (pre svega korisničkog imena) na serveru radi evidencije obavljeno je pozivom metode *CoQueryClientBlanket()* (Sl. 4) u serverskoj aplikaciji, dok je za preuzmanje oznake, tj. privilegija klijenta, za prisput datoteci korišćena *CoImpersonateClient()* funkcija (Sl. 6).



Sl. 3. Opšti prikaz preuzimanje klijentskih ovlašćenja

```
*ppszClientUsername = NULL;

hr = CoQueryClientBlanket(0,0,0,&nAuthnLevel,0,&hPrivs,0);

// Allocate memory
LPWSTR psz = AllocString((wchar_t*)hPrivs);

try
{
    *ppszClientUsername = AllocStringCOM(psz);
}
catch(...)
{
    FreeString(psz);
    throw;
}

FreeString(psz);
```

Sl. 4. Primer preuzimanja korisničkog imena logovanog klijenta



Sl. 5. Opšti prikaz preuzimanja klijentske oznake i „predstavljanje servera kao klijent”

```
HRESULT hr = CoImpersonateClient();

*pbFileOpenAllowed = TRUE;

FILE* fTestClientPermissions;
fTestClientPermissions = fopen(pszFileName,pszMode);

if ( fTestClientPermissions == NULL )
    *pbFileOpenAllowed = FALSE;
else
    fclose(fTestClientPermissions);

return hr;
```

Sl. 6. Primer ispitivanja prava pristupa datoteci na osvonu klijentske oznake

Podešavanja vezana za autentifikaciju, odn. povezivanje klijenata na server obavljena su globalno pomoću sistemkih administrativnih aplikacija (*dcomcnfg* za podešavanje prava pristupa serveru, kao i *compmgmt* za podešavanje prava pristupa udajenom računaru od strane nekog domenskog korisnika).

Prava pristupa datoteci (autorizacija) su takođe podešavana globalno sistemskim alatima *Windows*-a.

Reuzultati ispitivanja aplikacije (tačnije deo programskog koda) su primenjeni i na *UserServer*-u, DMS *server*-skoj aplikaciji za autentifikaciju i proveru prava DMS korisnika, gde su ta prava definisana u bazi podataka ili u aktivnom diretorijumu.

3. CORBA

Sa ciljem da se premosti jaz između programa napisanih na različitim programskim jezicima, koji se izvršavaju na računarima sa različitim operativnim sistemima, koji su povezani preko različitih mrežnih tehnologija korišćenjem različitih mrežnih protokola, nastao je OMG-ov (*Object Management Group*) standard CORBA (*Common Object Request Broker Architecture*). Uz pomoć CORBA standarda klijent može pozvati metodu serverskog objekta, koji može biti na istom računaru ili na istoj mreži. ORB prihvata poziv i preuzima odgovornost za pronalaženje objekta koji implementira dati zahtev, prosleđivanje parametara metode i vraćanje zahtevanih rezultata. Za poziv metode nekog objekta klijent ne mora da zna gde se traženi objekat nalazi (njegovu lokaciju na mreži) niti programski jezik na kom je implentiran niti operativni sistem na kome radi, već samo spregu (*interface*) objekta. Za specifikaciju tih sprega kojim se CORBA objekti predstavljaju na mreži, CORBA koristi IDL (*Interface Description Language*) koji se kompaliranjem preslikava na odgovarajući programski jezik. Isti princip komunikacije objekata preko sprega koristi i DCOM, naravno, napisanih *Microsoft*-ovom verzijom IDL-a.

3.1 OPŠTI SIGURNOSNI MEHANIZMI

Da bi sigurna komunikacija bila uopšte moguća, klijent i ciljani objekat moraju biti u stanju da koriste bar jedan od uobičajnih (opštih) sigurnosnih mehanizama. *CORBA Security Service Specification* definiše generalni model koji opisuje koncepte, terminologiju i arhitekturu referntnog sigurnosnog modela koji treba implementirati. Ova specifikacija obuhvata: identifikacija i autentifikacija principala, autorizacija (*access control*), revizija (*auditing*), intengritet (*integrity*), poverenje (*confidentiality*) i neporecivost (*non-repudiation*, opcioni paket). *Security Service* definiše, tj. razlikuje dva nivoa funkcionalnosti, tj. sigurnosnih API-a (*Application Programming Interface*):

- *Security Level 1* – aplikacija nije „svesna" sigurnosnih mehanizama,
- *Security Level 2* – aplikacija je „svesna" sigurnosti i sama kontroliše mehanizam sigurnosti.

CSI (*Common Secure Interoperability*) paket, kao deo *CORBA Security Services*-a, razlikuje tri nivoa interoperabilnosti koju implementacije usluga obezbeđuju. Ovim nivoima, koji podržavaju različite stepene delegacije ovlašćenja, su određeni sigurnosni mehanizmi koji se koriste za uspostavu sigurnosne asocijacije između inicijatora i ciljanog objekta. Nivoi interoperabilnosti:

- *CSI level 0, identity based policies without delegation* – Uspostavljanje sigurnosne komunikacije zasniva se na jednom atributu identiteta ovlašćenja (*credentials*) principala. Ne postoji mogućnost delegacije ovog atributa nekom drugom objektu ili iskorišćenje za zahtev neke druge usluge od strane tog objekta.
- *CSI level 1, identity based policies with unrestricted delegation* – Uspostavljanje sigurnosne komunikacije zasniva se na jednom atributu identiteta ovlašćenja principala. Postoji mogućnost delegacije ovog atributa, ali se ona ne može kontrolisati.
- *CSI level 2, identity and privilege based policies with controlled delegation* - Uspostavljanje sigurnosne komunikacije zasniva se na atributima identiteta ovlašćenja principala. Na ovom nivou omogućeno je, pored atributa identiteta, prenošenje i atributa privilegija. Delegacija ovih atributa je moguća i može se kontrolisati.

3.2 AUTENTIFIKACIJA I AUTORIZACIJA

Prilikom logovanja korisnika na računar, tj. operativni sistem, CORBA komponenta *PrincipalAuthenticator* uspostavlja skup sigurnosnih atributa identiteta koji predstavljaju ovlašćenja principala.

Uspostavljanje sigurnosne asocijacije između klijentskog programa i ciljanog objekta uključuje uspostavu poverenja o identitetu komunikacionih partnera (autentifikacija korisnika ili međusobna autentifikacija). Niti klijentski program niti ciljni objekat ne učestvuju direktno u procesu uspostave poverenja, tj. autentifikacije, već to za njih rade CSS (*Client Security Service*) i TSS (*Target Security Service*), pa se može nazvati autentifikacijom na nivou ORB-a. Ako postoji mogućnost da ciljna aplikacija „zna" identitet klijenta koji je pozvao njenu uslugu, onda se to može nazvati autentifikacijom na nivou aplikacije.

3.2.1 SAS PROTOKOL

SAS (*Security Attribute Service*) protokol se nalazi iznad bilo kog sigurnosnog mehanizma (protokola) na transportnom nivou i obezbeđuje dodatnu autentifikaciju klijenta, mehanizam delegacije i prenos ovlašćenja (autorizacione oznake). Ovaj protokol podeljen je na dva nivoa:

1. Nivo autentifikacije (*Authentication Layer*) služi da se izvrši autentifikacija klijenta na serveru kada to nije moguće na transportnom nivou zbog razlika u uspostavljanju sigurne komunikacije, tj. sigurnosnim mehanizmima, sa klijentske i strane poslužioca. Poruke SAS protokola se prenose unutar zaglavlja GIOP (*General Inter-ORB Protocol*) poruke i time je izbegnuta zavisnost od sigurnosnih protokola na transportnom nivou.
2. Nivo sigurnosnih atributa (*Security Attribute Layer*) klijent koristi da bi poslao svoja ovlašćenja (atribute o identitetu i pravima pristupa) objektu servera radi autorizacije i delegacije.

CSS započinje klijent-server komunikaciju slanjem TSS-u poruke *EstablishContext* u čijim poljima se nalaze sigurnosni atributi klijenta. U ovoj poruci klijent specificira:

- autorizacionu oznaku koja sadrži privilegije klijenta na objektu poslužioca (*authorization_token*),
- identifikacionu oznaku koja sadrži opis (identifikaciju) klijenta pri autorizaciji (*identity_token*),

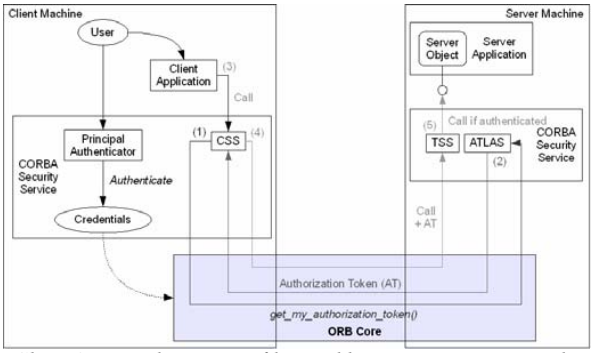
- autentifikacionu oznaku koja sadrži atribut identiteta i način tumačenja tog identiteta (*client_authentication_token*).

U slučaju da se u *EstablishContext* poruci ne nalazi autentifikaciona oznaka, TSS preuzima identitet klijenta iz transportnog nivoa. Ako je autentifikacija klijenta bila uspešna, TSS šalje CSS-u poruku *CompleteEstablishContext*, a u suprotnom poruku *ContextError*. Posle dobijanja poruke o uspešnoj autentifikaciji sledi slanje zahteva za uslugom kroz poruku *MessageInContext* čiji sadržaj TSS šalje serverskoj aplikaciji.

3.2.2 ATLAS

ATLAS (*Authorization Token Layer Acquisition Service*) specifikacija definiše postupak za akviziciju autorizacione oznake kojom se klijentu omogućava poziv usluge ciljnog objekta (Sl. 7). Ovim je rešen problem slanja klijentskih ovlašćenja koje će ciljani objekat razumeti.

Autorizaciona oznaka, koja se šalje CSIv2 protokolom, pored informacija o privilegijama klijenta sadrži i identifikacionu oznaku. Prema tome serverska aplikacija na osnovu autorizacione oznake može da sazna identitet klijenta čime je omogućena autentifikacija na nivou aplikacije kao i „predstavljanje" poslužioca kao klijent drugom poslužiocu. To „predstavljanje" u suštini znači delegiranje identiteta klijenta, tj. prosleđivanje atributa identiteta iz autoruzacione oznake ATLAS-u drugog poslužioca. ATLAS tog poslužioca na osnovu identiteta vraća novu autorizacionu oznaku koja sadrži prava pristupa klijenta uslugama tog drugog objekta poslužioca.



Sl. 7. Opšti prikaz autentifikacije klijenta primenom oznake pri pozivu metode na ciljnom objektu

4. ZAKLJUČAK

I DCOM i CORBA kao protokoli za komunikaciju *software*-skih komponenti u distribuiranim sistemima pružaju opšte mehanizme zaštite:

- proveru identiteta korisnika (autentifikaciju),
- proveru prava korisnika (autorizaciju)

što odgovara potrebama sistema, odn. zahtevima korisnika. Ovi mehanizmi u DCOM-u imaju svoju implementaciju, što je i pokazano primerom u poglavlju 2.3, dok u CORBA-i nepostoji (tačnije nije pronađena) stabilna *open source* implementacija mehanizama zaštite za aplikacije pisane na C++ programskom jeziku.

Od postojećih *open source* CORBA rešenja jedino TAO (*The ACE ORB*) i MICO (*MICO is CORBA*) ističu postojanje implementacije *security service*-a. Ali ispitivanjem njihovih najnovijih verzija (*TAO 1.5a* i *MICO 2.3.12*) zaključeno je da TAO sadrži samo kostur implementacije, dok je implementacija MICO *security service*-a nepotpuna.

```
SecurityLevel3::TargetCredentials_ptr
TAO::SL3::SecurityManager::get_target_credentials (CORBA::Object_ptr)
  ACE_THROW_SPEC ((CORBA::SystemException))
{
  throw CORBA::NO_IMPLEMENT ();
}
```

Sl. 8. Prikaz kostura implementacije security service-a u TAO-u

Teoretski CORBA bi mogla biti bolje rešenje za integraciju *Security Service*-a, pre svega zbog svoje *cross-platform* prirode, interoperabilnosti (pre svega različitih mehanizama zaštite) i dobre (mada generalno gledano, suviše složene) *security* specifikacije. Međutim, problem interoperabilnosti različitih operativnih sistema čini implementaciju sigurnosnih mehanizama u CORBA-i veoma složenom (ponekad čak i nemogućom). Ovo je sigurno jedan od razloga nepostojanja stabilne implementacije mehanizama zaštite i podstrek za dalja istraživanja.

LITERATURA

[1] Keith Brown, *Programming Windows Security*, Addison-Wesley, November 2000.

[2] OMG, *Security Service Specification*, Version 1.8, http://www.omg.org/docs/formal/02-03-11.pdf, March 2002.

[3] OMG, *Authorization Token Layer Acquisition Service (ATLAS) Specification*, Version 1.0, http://www.omg.org/docs/formal/02-10-01.pdf, October 2002.

[4] OMG, *Common Object Request Broker Architecture (CORBA)*, Version 3.1, Part 2: CORBA Interoperability, http://www.omg.org/docs/formal/08-01-04.pdf, January 2008.

Abstract – This paper presents fundamentals of distributed security system, i.e. authentication and authorization, and its implementation in DCOM and CORBA, through the functional analysis of their security measures. The primary goal is to demonstrate how authentication and authorization works in DCOM and CORBA through acquiring and delegation of client credentials.

AUTHENTICATION AND AUTHORIZATION IN DCOM AND CORBA BASED DISTRIBUTED SYSTEMS

Uglješa Novak, Srđan Orlović, Nikola Radovanović

UPRAVLJANJE VETROELEKTRANOM U SKLADU SA ZAHTEVIMA MREŽE

WIND PLANT CONTROL ACCORDING TO GRID CODE

D. Jerkan, V. Katić, Z. Ivanović, M. Vekić, Fakultet Tehničkih Nauka, Novi Sad

Oblast – *ELEKTROTEHNIKA I RAČUNARSTVO*

Kratak sadržaj – *U radu predstavljeni i upoređeni "zahtevi mreže" za priključenje vetroelektrana iz različitih elektroprivreda. Analizirane su situacije pri poremećajima napona u mreži, a glavni zahtev je da se i dalje prenosi maksimalna moguća snaga vetra. Postavljen je matematički model vetroelektrane i izvršene simulacije u programu Matlab. Razvijeni su odgovarajući odgovori upravljačkog algoritma u cilju održanja sistema unutar "zahteva mreže".*

Abstract – *The paper gives an overview and comparison of grid codes in different countries. The cases with disturbances in network are analyzed using maximum power demand as a criterion. The mathematical model is set and simulation in Matlab software has been performed. The special control algorithms have been developed to deal with such disturbances.*

Ključne reči*: zahtevi mreže, upravljanje vetroelektranom, propadi napona, učestanost*

1. UVOD

Tendencije koje nameću ekološku, a naravno i ekonomsku opravdanost sve veće upotrebe alternativnih, prvenstveno obnovljivih izvora električne energije, među kojima energija vetra uzima ključno mesto, sa sobom povlače i neophodnost da se taj, sve veći udeo u ukupnoj proizvodnji električne energije što bolje i efikasnije uklopi u već postojeću infrastrukturu električnih mreža. U ranijem periodu, kada je količina energije dobijene iz vetra bila mala u odnosu na ukupno proizvedenu električnu energiju, postojala je i praksa da se vetroelektrane isključe iz sistema u slučaju pojave nekog mrežnog poremećaja. Danas, a pogotovo u budućnosti, nametaće se sve strožiji zahtevi koje vetroelektrane moraju da ispune u slučaju pojave takvih poremećaja. Takvi trendovi će jednostavno postati nužni, jer elektroenergetski sistemi neće imati dovoljne proizvodne kapacitete koji bi bili u stanju da nadomeste simultani ispad velikog broja vetroelektrana, pa ujedno i jednovremeni gubitak električne energije koju one injektuju u elektroenergetski sistem, što bi moglo da dovede do kolapsa istog. Tu se prvenstveno misli na neophodnost njihovog ostanka u pogonu onoliko dugo vremena koliko je to propisano, naravno u zavisnosti od vrste samog poremećaja. Pod poremećajima se ovde podrazumevaju propadi napona, kao posledica pojave kratkih spojeva u mreži i fluktuacije mrežne učestanosti usled naglog opterećivanja ili rasterećivanja potrošnjom elektroenergetskog sistema na koji su vetroelektrane priključene. Ovi zahtevi su definisani u posebnim tehničkim propisima koji se nazivaju i " zahtevi mreže" (Grid Code). Cilj rada je iznalaženje optimalnog upravljanja i postizanje stabilnog rada vetroelektrane u gorenavedenim režimima. U radu su grafički predstavljeni zahtevi mreže različitih elektroprivrednih firmi (E.ON i Eltra iz Nemačke, kao i drugi) i dato je njihovo poređenje. Zatim je postavljen matematički model vetroelektrane i dat je njegov simulacioni model u programskom paketu Matlab. Poseban akcenat je stavljen na detaljniji opis onog dela upravljačkog algoritma,koji implementira dvojaku strujnu regulaciju mrežnog konvertora (DVCC),kao i PLL (Phase-Locked Loop), koji obezbeđuje efikasnu i robusnu sinhronizaciju sa mrežom, nezavisno od toga koji je tip poremećaja nastupio u njoj, što u mnogome povećava efikasnost i stabilan rad vetroelektrane. Kao potvrda ovim tvrdnjama, priloženi su i rezultati simulacija sprovedenih nad datim modelima, u kojima su prikazani odzivi karakterističnih veličina same vetroelektrane i električne mreže, sa naglaskom na režime električne mreže sa poremećajima.

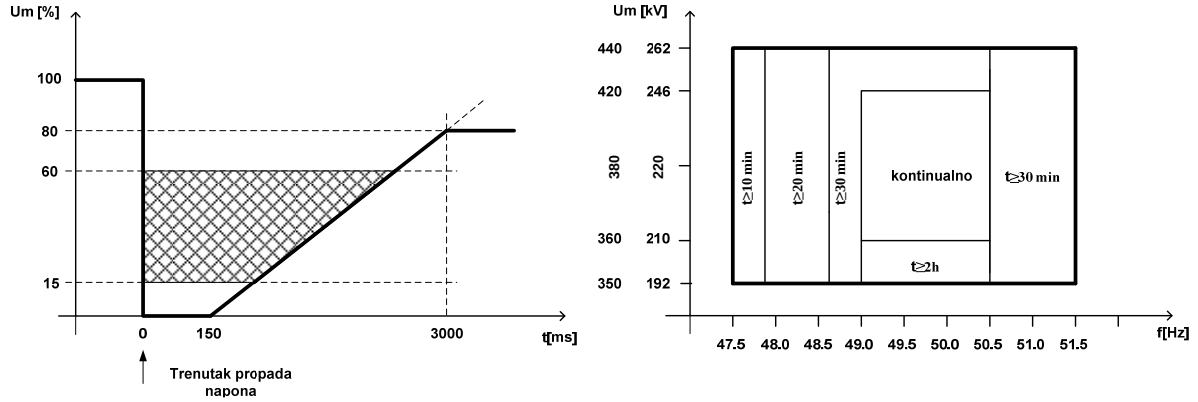
2. ZAHTEVI MREŽE ("Grid Code")

U evropskim zemljama koje predstavljaju lidere u proizvodnji električne energije iz vetroelektrana (Nemačka, V.Britanija, Danska, Španija i druge) uveliko su razvijeni i prisutni strogi tehnički propisi koji regulišu ponašanje vetroelektrana u skladu sa zahtevima mreže. U njima su definisani vremenski intervali u kojima vetroelektrane moraju i dalje biti priključene na električnu mrežu, u zavisnosti od toga kakav je tip poremećaja u mreži nastao. Na Slici 1. dat je prikaz zahteva mreža nemačkih elektroprivrednih firmi E.ON i Eltra, gde je na Slici 1. a) imamo grafički prikaz zahteva mreže E.ON-a [1] za trajanjem rada vetroelektrane u slučaju propada mrežnog napona ("Fault Ride Through"). Na Slici 1. b) prikazan je zahtev mreže E.ON-a i Eltre koji se tiču varijacija mrežne učestanosti, za različite naponske nivoe, uz propisane vremenske intervale u kojima vetroelektrane moraju ostati priključene na električnu mrežu. Na Slici 2. prikazan je zahtev mreže E.ON-a koji se odnose na propade mrežnog napona uzrokovanim kvarom u neposrednoj blizini generatora. Na Slici 2 b) prikazan je zahtev mreže E.ON-a i Eltre koji se odnosi na propade napona koji se tiču farme vetroelektrana. Na Slici 3 a) i b) respektivno su dati grafički prikazi zahteva mreže škotske [2] i irske [3] elektroprivredekoji se tiču propada napona. Važno je primetiti da u svim dole prikazanim zahtevima mreže nema velikih međusobnih razlika, što je naravno i očekivano, jer se teži njihovom unificiranju. Potrebno je još napomenuti da će ovi zahtevi postajati sve strožiji u budućnosti, kako bude rastao broj i ukupna instalisana električna snaga vetroelektrana u elektroenergetskom sistemu. Doći će se i do tačke kada će se one morati tretirati kao konvencionalne elektrane. U ovom radu se nećemo baviti svim zahtevima mreže, već samo sa dva gorenavedena, ali treba napomenuti da postoje i zahtevi obuhvataju i mogućnost varijacije proizvodnje reaktivne energije vetroelektrana, odnosno rad sa promenljivim faktorom snage, kao i propisani nivoi harmonijskog izobličenja koji vetroelektrane smeju da injektuju u sistem, kao i drugi.

Napomene:

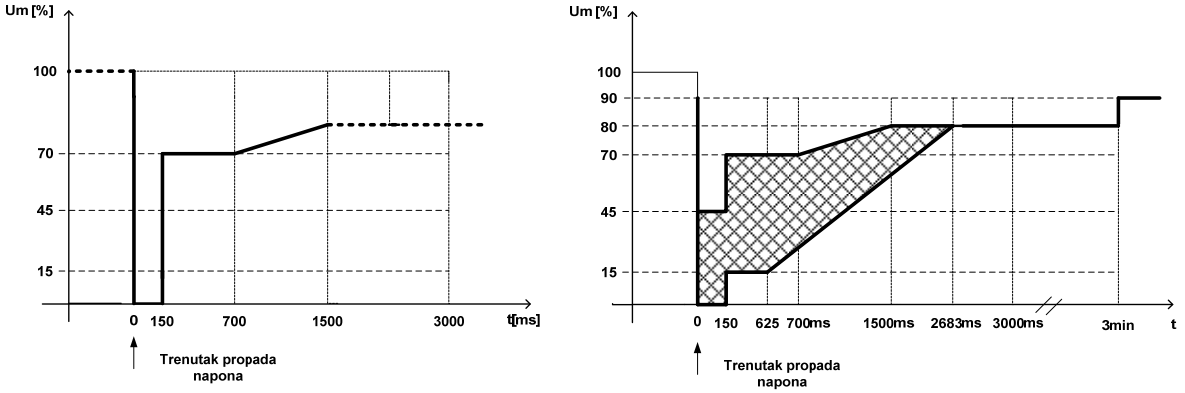
a) Rad je proistekao iz diplomskog-master rada Dejana Jerkana. Mentor je bio prof.dr Vladimir Katić.

b) Rad je prethodno publikovan na konferenciji JUKO-CIRED, Vrnjačka Banja, Oktobar 2008.

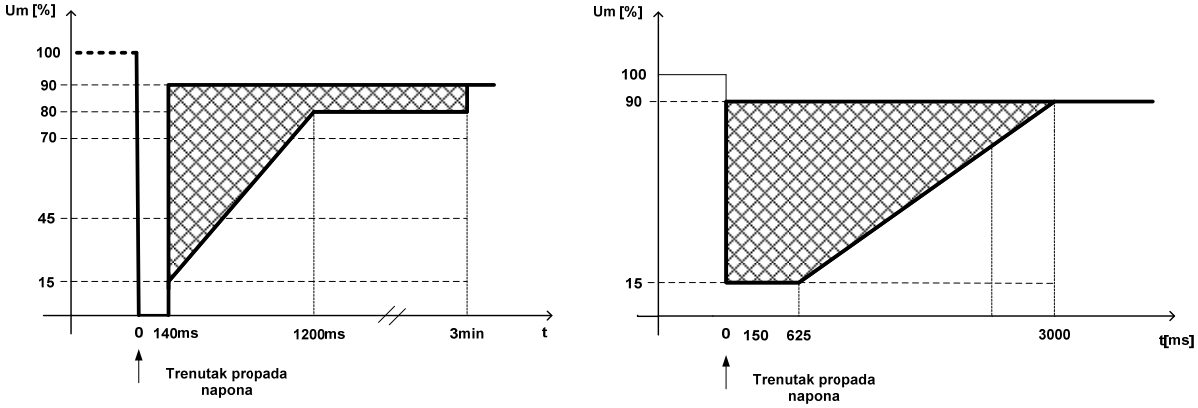


a) b)

Sl. 1. *a) Zahtev mreže E.ON-a, koji se tiče propada napona; b) Zahtev mreže E.ON-a i Eltre koji se tiče varijacija učestanosti*



Sl. 2. *Zahtevi mreže za propade napona: a) E.ON-a u blizini samog generatora; b) E.ON-a i Eltre za farme vetroelektrana*



Sl. 3. *a)Zahtev mreže škotske elektroprivrede za propade napona; b) Zahtev mreže irske elektroprivrede za propade napona*

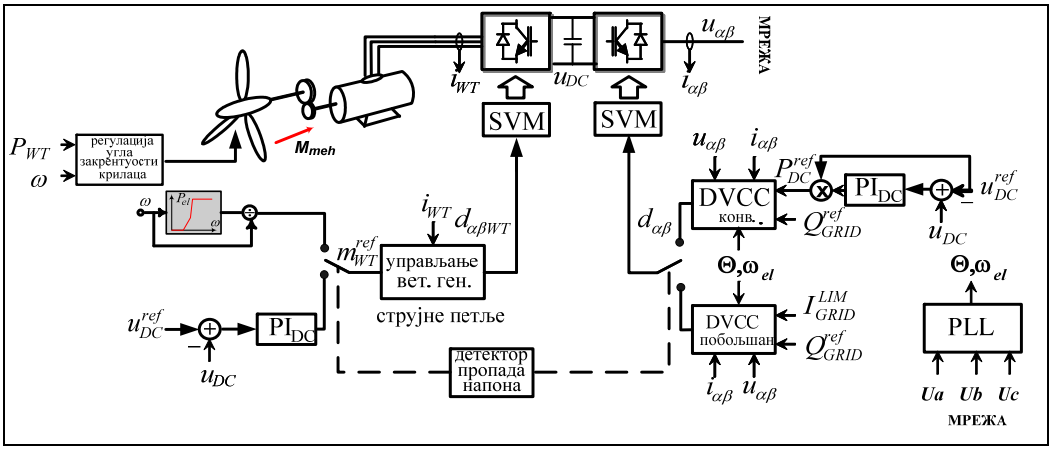
3. MODEL VETROELEKTRANE

Pogon vetroelektrane sastoji se od vetrogeneratora i dvostrukog pretvarača. Vetroturbina zahvata silu vetra. Kao rezultat toga javlja se moment, koji se preko reduktora (multiplikatora) za prilagođavanje (povećavanje) brzine turbine brzini generatora, prenosi na vratilo asinhronog generatora. Dvostruki pretvarač se sastoji iz dva pretvarača: pretvarač prema generatoru (ispravljač) i pretvarač prema mreži (invertor), koji reguliše protok aktivne i reaktivne snage prema mreži. Ova dva pretvarača su povezana jednosmernim međukolom. Jednosmerno međukolo frekvencijski raspreže elektroenergetski sistem od sistema vetroelektrane [4]. Asinhrona mašina je vektorski upravljana u cilju nezavisnog upravljanja momentom i fluksom [5]. Posebnu pažnju treba obratiti na regulacionu strukturu pretvarača prema mreži, koja obuhvata konvencionalnu i modifikovanu dvojaku strujnu regulaciju (DVCC), kao i blok koji obezbeđuje sinhronizaciju sa mrežom (PLL).

4. REGULACIONA STRUKTURA UPRAVLJANJA PRETVARAČEM PREMA MREŽI

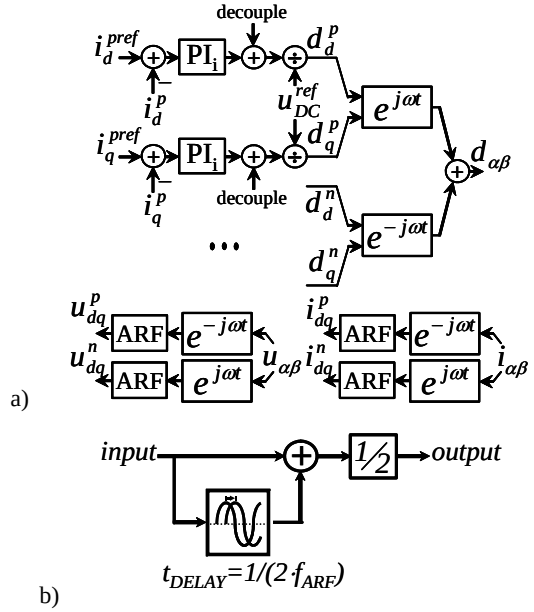
U ovom poglavlju biće predstavljene regulacione strukture upravljanja pretvaračem prema mreži u uslovima nesimetričnih propada mrežnog napona.Kao što je već navođeno u prethodnom poglavlju, vetroelektrane moraju da se povinuju zahtevima mreže koji se tiču propada napona, a kako je poznato da su u električnoj mreži daleko najviše zastupljeni upravo nesimetrični propadi napona, jasno je da je neminovno razvijati i usavršavati upravo takve algoritme upravljanja vetroelektranama. Postojeća rešenja se zasnivaju na tzv. DVCC-u [6]. Ovaj metod se zasniva na regulaciji direktne i inverzne komponente struja, što omogućava prenos sve raspoložive snage na mrežnoj učestanosti, uz suzbijanje oscilacija na dvostrukoj učestanosti. Veliki nedostatak ove metode predstavlja nepostojanje nadzora nad temenim vrednostima struja, koje u slučaju dubljih propada napona mogu dostići vrednosti i nekoliko puta veće od

nominalnih. Moguće rešenje ovog problema jeste predimenzionisanje samog pretvarača, što je ekonomski krajnje neisplativo rešenje. Drugo rešenje predstavlja ograničavanje izlaznih struja.To automatski znači i ograničavanje izlazne snage elektrane, a sa sobom povlači i oscilacije reaktivne snage tokom propada napona. Rešenje, odnosno algoritam primenjen u ovom radu omogućuje kako nadzor izlaznih struja, tako i eliminaciju oscilacija na dvostrukoj učestanosti aktivne i reaktivne snage [7]. Taj algoritam, pod nazivom Modifikovani DVCC se zasniva na nezavisnoj regulaciji komponenti struja pretvarača po d i q osi dobijenih transformacijom u pozitivno, odnosno negativno rotirajućem koordinatnom sisitemu, respektivno (i_d^p i i_q^p; i_d^n i i_q^n). Da bi se postiglo potpuno rasprezanje regulacionih struktura za pozitivne i negativne komponente struja, neophodno je eliminisati oscilacije na dvostrukuj učestanosti koje se pojavljuju kao posledica primene transformacije obrtanja na simetrične komponente struja.



Sl. 4. *Kontrolna struktura vetroelektrane*

Detaljniji matematički model Modifikovanog DVCC-a, kao i proračun parametara odgovarajućih PI regulatora i referentnh vrednosti struja invertora po obe ose može se naći u [8].]. Na Slici 5 a) prikazana je regulaciona struktura Modifikovanog DVCC-a, a na Slici 5 b) prikazana je struktura antirezonantnog filtra (ARF). Na Slici 6 prikazani su karakteristični odzivi vetroelektrane dobijeni simulacijom u programskom paketu Matlab, u slučaju pojave propada napona tipa C. Pretpostavlja se da je napon u fazama *b* i *c* opao na 40% nazivne vrednosti.



a)

b) $t_{DELAY}=1/(2f_{ARF})$

Sl. 5. *a) Regulaciona struktura trenutne regulacije snage; b) Dijagram antirezonantnog filtra (ARF)*

5. MODEL PLL-a

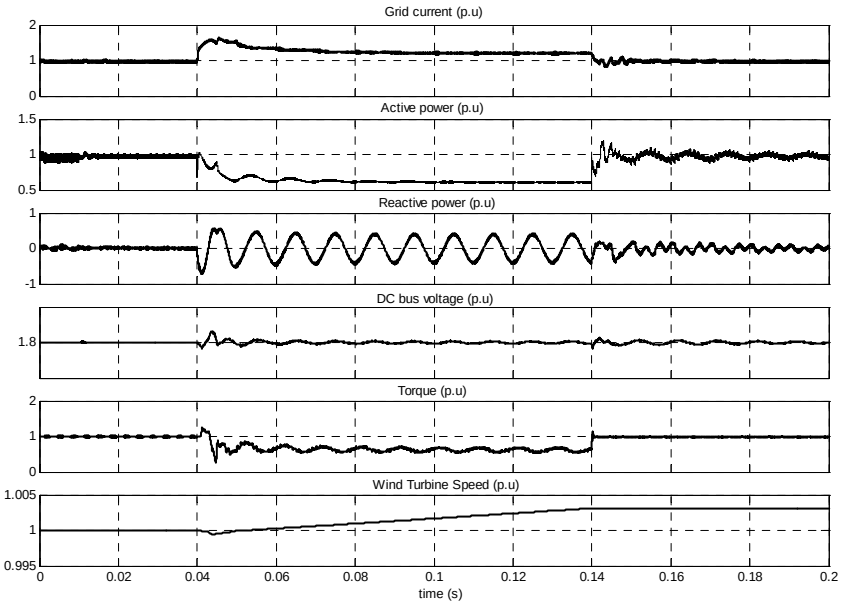
U slučaju pojave poremećaja u električnoj mreži, koji u sebi obuhvata promenu faznog stava mrežnih napona ili čak varijaciju mrežne učestanosti, od suštinskog značaja je postići kvalitetnu i robusnu sinhronizaciju pretvarača vetroelektrane prema mreži. Ako to nije slučaj, može doći do pojave veoma velikih struja izjednačenja, koje teku između pretvarača i mreže, koje mogu rezultovati oštećenjima samog pretvarača, te su stoga nedopustive. U ovom poglavlju je prikazano jedno rešenje koje obezbeđuje efikasnu sinhronizaciju sa mrežom, nezavisno od toga da li je nastupio poremećaj koji u sebi obuhvata čak i propade napona i varijacije učestanosti istovremeno. Osnovni deo prikazanog PLL-a predstavlja tzv. SOGI-QSG (Second Order Generalized Integrator for Quadrature Signals Generation) [8], generalizovani integrator drugog reda za generisanje signala fazno pomerenih za 90°, prikazan na Slici 7. Prenosne funkcije datog bloka date su sa (1) i (2). Ako je ulazni signal sinusoida učestanosti ω koja se poklapa sa rezonantnom učestanošću ω', tada se kao izlazi SOGI-QSG dobijaju naponi jednaki po amplitudi, a fazno pomereni za 90°, nezavisno od vrednosti pojačanja *K* što se da zaključiti iz (3) i (4), na koje se izrazi (1) i (2) mogu svesti u gorenavedenom slučaju.

$$D(s)=\frac{v'}{v}(s)=\frac{K\omega'\omega}{s^2K\omega's+\omega'^2} \tag{1}$$

$$Q(s)=\frac{qv'}{v}(s)=\frac{K\omega'^2}{s^2K\omega's+\omega'^2} \tag{2}$$

$$qv'=Qv\begin{cases}|Q|=\dfrac{\omega'}{\omega}|D|\\ |Q|=|D|-\dfrac{\pi}{2}\end{cases} \tag{3}$$

$$v'=\begin{cases}|D|=\dfrac{K\omega\omega'}{\sqrt{(K\omega\omega')^2+(\omega^2-\omega'^2)^2}}\\ \left[D=\tan^{-1}\left(\dfrac{\omega'^2-\omega^2}{K\omega\omega'}\right)\right]\end{cases} \tag{4}$$



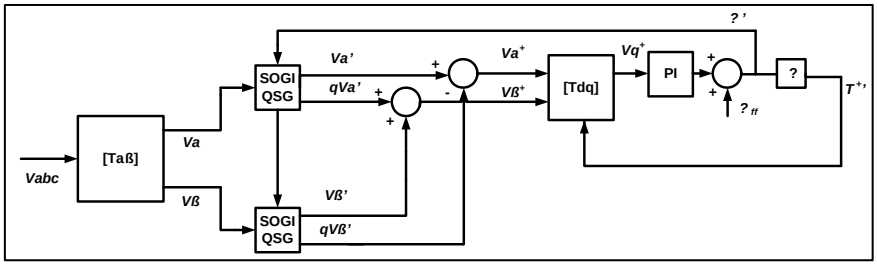
Sl. 6. *Odzivi karakterističnih veličina vetroelektrane u slučaju pojave propada napona tipa C*

Međutim, ako se vrednosti ω i ω' ne poklapaju, tada dolazi kako do odstupanja izlaza SOGI-QSG i po amplitudi i po faznom pomeraju, koji više ne iznosi 90°. Ako bismo kao ulaz dva ovakva integratora dovodili α i β komponente mrežnog napona, respektivno i njihove izlaze povezali kako je prikazano na Slici 8, na izlazu sabirača bismo dobili modifikovane vrednosti αβ komponenti napona iz kojih bi se jednostavno mogao dobiti fazni stav prvog harmonika mrežnog napona,uz uslov ω=ω':

$$\Theta^{+\prime} = \tan^{-1}\frac{V_\beta^+}{V_\alpha^+} \tag{5}$$

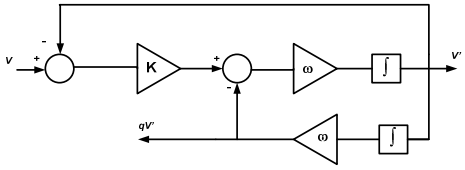
Ako je, međutim, došlo do varijacije mrežne učestanosti, tako da više ne važi jednakost učestanosti osnovnog harmonika mrežnog napona i rezonantne učestanosti SOGI-QSG, ovaj metod bi nam dao netačne rezultate. Modifikovane vrednosti αβ komponenti koji se dobijaju na izlazu dvostrukog SOGI-QSG bi u sebi sadržale više harmonike, čiji su izrazi za moduo i fazni stav dati u (6).

Važno je napomenuti,da iako imamo nejednakost učestanosti prvog harmonika mrežnog napona i rezonantne učestanosti SOGI-QSG, važi izraz (7), koji je takođe nezavisan od vrednosti pojačanja K. Bitna osobina ovog sklopa je i da se on ponaša kao NF filtar za simetrične komponente pozitivnog redosleda mrežnog napona. Ta osobina je iskorišćena, da uvođenjem zatvorene povratne sprege, pomoću PI regulatora, izvršimo dinamičku korekciju rezonantne učestanosti i na taj način ponovo obezbedimo pretpostavke iz kojih sledi (5), tj. obezbedimo efikasnu sinhronizaciju sa mrežom. Struktura koja omogućuje rešenje ovog problema, pod nazivom DSOGI-PLL (Dual Second Order Generalized Integrator Phase Locked Loop), prikazana je na Slici 8. Blokovi T[αβ] i T[dq] predstavljaju blokove za transformaciju mrežnih napona iz abc domena u αβ domen (8), odnosno transformaciju iz αβ domena u dq domen (9).



Sl. 7. *SOGI-QSG*

$$V_\alpha^+ = P^n V_\alpha^n \begin{cases} |P^n| = \dfrac{K\omega'}{2}\sqrt{\dfrac{(n\omega+\omega')^2}{(Kn\omega\omega')^2+(n^2\omega^2+\omega'^2)^2}} \\ \angle P^n = sgn(n)\tan^{-1}\left(\dfrac{\omega'^2-n^2\omega^2}{Kn\omega\omega'}\right)-\dfrac{\pi}{2}(1-sgn(n^2\omega+n\omega')) \end{cases} \tag{6}$$



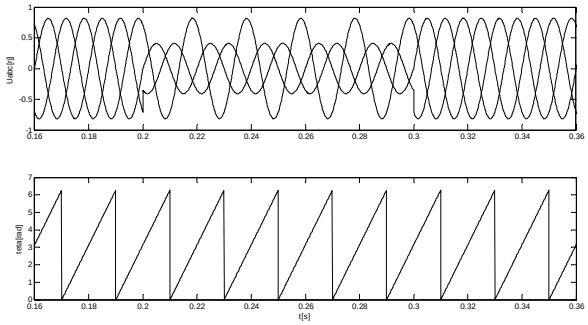
Sl. 8. *DSOGI-PLL*

$$|V_\beta^+| = |V_\alpha^+| \,;\, \angle V_\beta^+ = \angle V_\alpha^+ - sgn(n)\frac{\pi}{2} \tag{7}$$

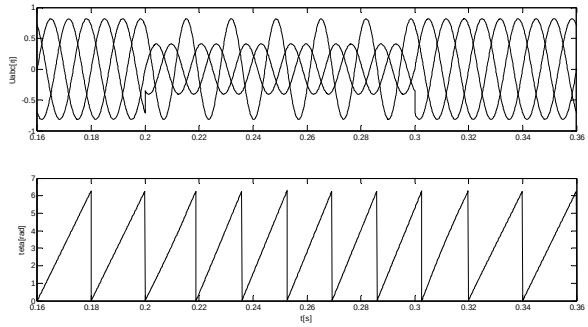
$$[T_{dq}] = \begin{bmatrix} \cos\Theta^+ & \sin\Theta^+ \\ -\sin\Theta^+ & \cos\Theta^+ \end{bmatrix} \tag{8}$$

$$[T_{\alpha\beta}] = \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \tag{9}$$

Vrednosti proporcionalnog, odnosno integralnog dejstva PI regulatora iznose K_p=2.22 i K_i=61.69, a postupak njihovog proračuna je dat u [9]. Na Slici 9 je prikazan odziv DSOGI-PLL u slučaju pojave propada mrežnog napona tipa C, gde su vrednosti napona u fazama b i c pale na 50% nazivne vrednosti. Vidi se da dati algoritam dobro estimira fazni stav mrežnih napona, bez obzira što je u mreži nastupio kratkotrajni propad napona. Na Slici 10 je prikazana još kritičnija situacija, gde je osim propada napona nastala i promena mrežne učestanosti.Ovde je, zbog preglednosti, načinjen prilično veliki skok mrežne učestanosti sa 50Hz na 60Hz. Vidi se da i u ovom slučaju algoritam dobro estimira fazni stav mrežnog napona.



Sl. 9. *Rezultati simulacije DSOGI-PLL algoritma u slučaju pojave propada mrežnog napona tipa*



Sl. 10. *Rezultati simulacije DSOGI-PLL algoritma u slučaju propada napona tipa C i poremećaja učestanosti*

Za simulaciju upotrebljenog generatora korišćeni su realni parametri dobijeni od proizvođača. Uzet je primer asinhronog kaveznog vetrogeneratora snage 2MW. U nastavku su dati najvažniji parametri ovog generatora:
P_n=2000 kW; U_n=6kV; I_n=220,3A; p=4; cosφ = 0,9162; s_n=0,8172%; η_n=0,9534; J=430kgm^2; f_n=50Hz; R_s=0,12094Ω; R_r=0,140784Ω; X_{ls}=1,33774Ω; X_{lr}=0,98214Ω, X_m=54,58 Ω.

Za ostale veličine korišćeni su sledeći parametri:
Parametri električne mreže: R=0,1Ω; L=1000 mH.
Parametri jednosmernog međukola: C=100μF; U_{DC}=10,8kV.

6. ZAKLJUČAK

U radu su predstavljeni zahtevi mreže različitih elektro-privrednih firmi i dato je njihovo poređenje, uz zaključak da oni ne odsupaju mnogo jedni od drugih, što svakako ide u prilog njihovom olakšanom unificiranju u budućnosti. Zatim su prikazani poboljšani upravljački algoritmi koji obezbeđuju robusno upravljanje vetroelektranom, kao i efikasnu sinhronizaciju sa mrežom, u skladu sa analiziranim zahtevima mreže. Na taj način se raspoloživa energija vetra može i dalje injektovati u električnu mrežu, nezavisno od poremećaja u samoj mreži, što obezbeđuje optimalnu iskorišćenost same vetroelektrane.

LITERATURA

1. EON Netz, German TSO, "*Grid Code, High and extra high voltage*“, 2005.
2. Hydro-Electric Transmission Ltd., Scottish TSO, *"Scottish Grid Code*“, 2006.
3. ESB National Grid, Irish TSO, 2005, "*Wind Farm Power Station Grid Code Provisions*“
4. B.Bimal, “*Modern power electronics and AC drives”*, Prentice Hall, New York, 2002.
5. I.Boldea, S.Nassar, *"Electric Drives"*, Taylor & Francis, New York, 2006.
6. Z.Ivanović, M.Vekić, S.Grabić, V.Katić, 2006, “Control of Multilevel Converter Driving Variable Speed Wind Turbine in Case of Grid Disturbances”, 12th EPE-PEMC 2006, Portoroz, Slovenia, pp.1569-1573.
7. A.Mullane, G.Lightbody, R.Yacamini, "Wind-Turbine Fault Ride-Through Enhancement", IEEE Transact. on Power Systems, Vol.20, No.4, pp. 1929–37, Nov. 2005.
8. P.Rodriugez, R.Teodorescu, I.Candela, et all, "New Positive-sequence Voltage Detector for Grid Synchronization of Power Converters under Faulty Grid Conditions", IEEE PESC’06, pp. 1-7, June 2006.
9. S.Chung, *"A phase tracking system for three phase utility interface inverters," IEEE Trans. on Power Electronics*, Vol.15, pp.431-438, May 2000.

ELEKTRONSKI SISTEMI ZASNOVANI NA BLUETOOTH KOMUNIKACIJI

Dražen Gurović, *Fakultet tehničkih nauka, Novi Sad*
dgurovic@gmail.com

Sadržaj – *Cilj ovog rada jeste da se pokažu prednosti kao i načini sinhronzacije i komunikacije elektronskih sistema korišćenjem bluetooth komunikacija.*

1. UVOD

Elektronski sistemi današnjice obavaljaju kompleksne zadatke. Pošto je, u većini slučajeva, takav sistem nemoguće napraviti kao hardverski jedinstven uređaj pribjegava se dekompoziciji sistema na manje mikroračunarske sisteme i potom njihovim uvezivanjem dobija se jedinstven sistem upravljanja. Pod pojmom mikroračunarski sistem podrazumjeva se logički nezavisna cjelina koja upravlja ili kontroliše jednom ili više promjenjljivih elektronskog sistema. Najbolji primjer centralizovanog elektronskog sistema predstavlja inteligentna kuća. Ovaj sistem sastoji se od skupa mikroračunarskih sistema (kontrola osvjetljenja, kontrola temperature, video nadzor, alarmni sistem) koji su međusobno povezani i sinhronizovani u jednu logičku cjelinu.

Načini povezivanja mikroračunarskih sistema u cijelinu predstavljaju vrlo važan segment od kog zavisi brzina i efikasnost sistema. Jedan od načina njihovog povezivanja jeste *bluetooth*. *Bluetooth* je standard za bežični prenos podataka i govora, kratkog dometa, namjenjen za malu potrošnju i jeftine bežične komunikacije. Ovaj standard podržava *point-to-point* i *point-to-multipoint* aplikacije, a osnovna namjena mu je formiranje personalnih mreža, PAN (*Personal Area Networks*)

Cilj ovog rada je da se predstave prednosti Bluetooth načina komunikacije u odnosu na druge vidove komunikacije između mikroračunarskih sistema i predoče razlozi zbog čega je ona pogodnija za realizaciju kompleksnih elektronskih sistema kako i da se objasne načini komunikacija između istih.

2. OSNOVNI KONCEPTI SISTEMA

U osnovi elektronske sisteme upravljanja možemo podjeliti na :

1. Centralizovane sisteme
2. Decentralizovane sisteme

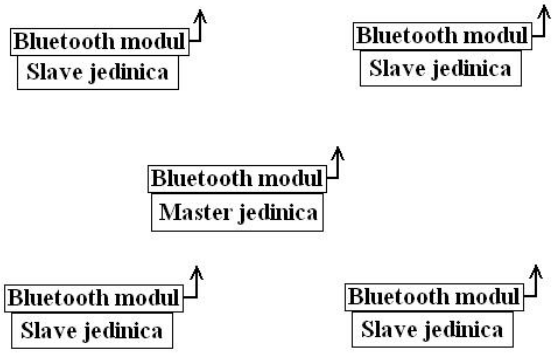
Kod centralizovanih sistema imamo SLAVE mikroračunarske module koji izvršavaju određeni dio zadatka, ali njihovo sinhronizovanje sa ostatkom sistema vrši glavni MASTER mikroračunarski modul. Mana ovakvih sistema jeste što se komunikacija između SLAVE modula vrši isključivo preko MASTER-a. Dakle MASTER modul mora da vodi računa o svim ostalim modulima, što ima za poslijedicu da on mora imati znatno veću procesorsku snagu u odnosu na ostatak sistema jer bi u suprotnom došlo do zagušenja. Prednost ovakvih sistema jeste jednostavno uvođenje novih SLAVE modula, reprogramiranje sistema, pošto je potrebno reprogramirati samo MASTER, jednostavan upravljački algoritam kao i trenutni uvid u kompletno stanje sistema.

Kod decentralizovanih sistema, mikroračunarski moduli su u potpunosti nezavisne cjeline, koje mogu da djeluju bez ostatka sistema, tj. bez glavnog modula. U ovom slučaju, modul će inicirati vezu samo sa onim modulima od kojih zavisi njegova regulisana veličina. Osnovna ideja ovakvih sistema jeste da se cjelokupna upravljačka hijerarhija, jedan MASTER i više SLAVE-ova, podijeli na dijelove, tj. da je svaki modul ravnopravan u sistemu. Na taj način dobijaju se inteligentni moduli koji mogu da interreaguju sa sredinom, tj. da u cilju finijeg podešavanja promjenjljive veličine , ukoliko postoji mogućnost, uzimaju više parametara, od ostalih modula sistema, za svoj rad.

U ovakvim sistemima nameće se pitanje koji vid komunikacija koristiti, tj. kako ih povezati. U slučaju da se koristi kabel, takvo rješenje je moguće ali jako nepraktično. Jedan od osnovnih razloga za to jeste instalacija novog modula u sistem. U centralizovanim sistemima potrebno je novi modul povezati sa MASER-om, što sa sobom povlači razne građevinske radove. U decentralizovanim sistemima njegova instalacija bi izazvala pravu pometnju, tj. novi modul je potrebno povezati sa svim ostalim modulima u sistemu. Drugi način povezivanja modula u sistemu jeste *wireless*, koji svakako predstavlja bolji i efikasniji nači i po pitanju instalacije novih modula kao i o njihovoj mobilnosti.

U daljem razmatranju predstavićemo cenralizovani sistem sa bluetooth komunikacijom, kao i navesti sve prednosti i mane istog.

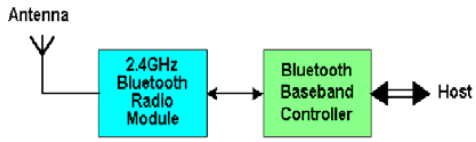
Sistem se sastoji od jednog glavnog modula, centralne jedinice i jedne ili više izvršno/perifernih jedinica. Arhitektura sistema se zasniva da periferne jedinice (SLAVE) obrađuju odgovarajuće podatke, vrše regulaciju promjenjljive sistema, dok za njihov sinhronizovani rad se koristi glavni modul (MASTER). U ovakvom sistemu, periferne jedinice ostvaruju konekciju samo sa glavnim modulom, primaju od njega instukcije za rad i šalju mu svoje trenutno stanje, status ili informacije koje se od iste zahtjevaju. Ovakva arhitektura prikazana je na slici 1.



Sl. 1. *Osnovni koncept sistema zasnovan na bluetooth komunikaciji.*

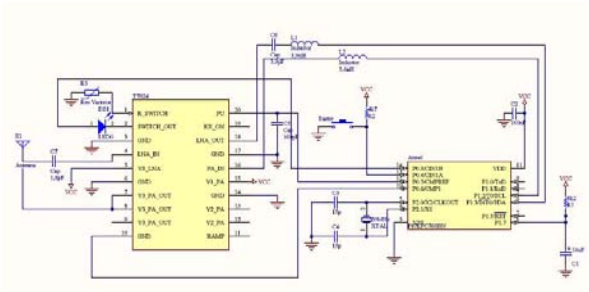
Napomene:
a) Rad je proistekao iz diplomskog-master rada Dražena Gurovića. Mentor je bio prof.dr Veljko Malbaša.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

MASTER i SLAVE su sastavljeni od dva osnovna bloka, bluetooth modula za komunikaciju i mikroprocesorskog bloka. Bluetooth modul je prikazan na slici 2.



Sl. 2. *Izgled bluetooth modula*

Za mikroprocesorski blok čini mikroprocesor at89s52 [4] sa svojim potrebnim periferijama. Veza između bluetooth i mikroprocesora se vrši odgovarajućim data i kontrolnim linijama što je prikazano na slici 3.



Sl. 3. *Povezivanje mikrokontrolera st89d52 sa bluetooth modulom*

3. NAČIN RADA SISTEMA

MASTER najprije pronalazi, definiše i uspostavlja vezu sa SLAVE-ovima koji predstavljaju odgovarajući mikroračunarski sistem. Ukoliko na području rada postoje još neki uređaji sa kojima se može uspostaviti veza ili je oni iniciraju sa MASTER-om, a pri tome nisu definisani u sistemu njihov zahtjev za komunikaciju se odbija. Nakon uspostavljanja veze sa svim SLAVE kontrolerima, završava se proces inicijalizacije mikroračunarskog sistema. Sada je sistem spreman i u potpunosti operativan.

Ukoliko prilikom inicijalizacije sistema MASTER nije uspostavio komunikaciju sa jedanim ili više SLAVE-ova, uspostavlja se veza sa dostupnim dijelovima sistema i nakon isteka odovarajućih vremenskih intervala nastavlja se sukcesivna potraga za nedostajućim SLAVE-ovima. Razlozi zbog kojih SLAVE ne može da se javi, po vremenskom kriterijumu, mogu biti kratkotrajni ili dugotrajni prekidi u komunikaciji. Uzroci kratkotrajnih prekida mogu biti inicijalizacija mikrokontrolera u SLAVE jedinici ili trenutna zauzetost, što ne predstavlja veliki problem. Dugotrajni prekidi u komunikaciji SLAVE sa MASTER kontrolerom je znatno ozbiljniji problem koji može nastati uslijed hardverskog kvara na SLAVE-u što je alarmantno stanje. MASTER kontroler će tražiti izostalog SLAVE unaprijed definisan vremenski interval i ukoliko ga ne bude pronašao potrebno je obustaviti rad sistema i signalizirati grešku.

Komunikacije MASTER-a sa ostalim dijelovima sistema zasnovana je da se najprije pošalje set odgovarajućih komandi, nakon čega prozvani SLAVE šalje potvrdu o uspješno primljenoj a potom i o statusu izvršavanja date komande. Na ovaj način se poboljšava stabilnost, sigurnost i operativnost elektronskog sistema sistema.

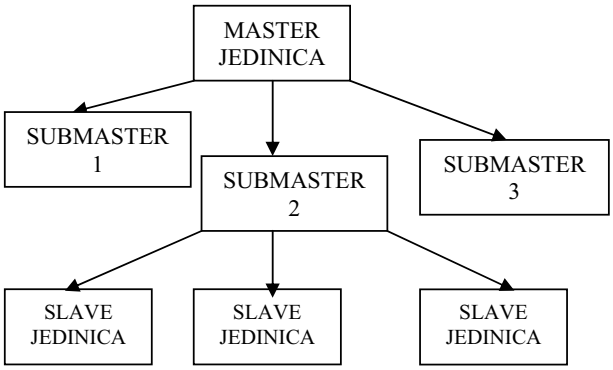
4. NEDOSTACI OVAKVOG SISTEMA

Osnovna mana u komunikacijama, ako se može tako nazvati, ovakvog sistema jeste nemogućnost da MASTER istovremeno uspostavi i održava vezu sa više od osam SLAVE jedinica. U cilju prevazilaženja ovog problema imamo dva načina rješavanja:

1. Metodom binarnog stabla
2. Metodom vremenskog multipleksiranja

Metoda binarnog stabla :

Osnovna ideja ove metode jeste da MASTER komunicira sa odgovarajućem brojem subMASTER kontrolera, do osam, kako bi preko njih mogao upravljati i imati konstantnu vezu sa svim dijelovima sistema. Ovakav tip rješenja prikazan je na slici 4.



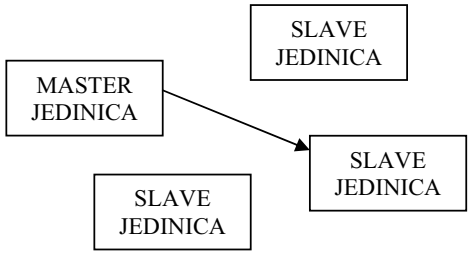
Sl. 4. *Metoda binarnog stabla*

Prednost ovog rješenja jeste što MASTER ima neprekidan uvid u stanje sistema, može da interveniše trenutno. Mana ovakvog rješenja jeste usložnjavanje sistema tj. uvođenje novih subMASTER jedinica, čime se usporava brzina protoka podataka MASTER – SLAVE, povećavanje cijene samog sistema.

Ovakva arhitektura sistema se koristi u slučaju kada je neophodna neprekidna komunikacija MASTER-SLAVE, kao npr. sistemi gdje SLAVE-ovi predstavljaju senzorske uređaje za mjerenje brzo promjenjljivih signala.

Metoda vremenskog multipleksiranja:

Drugo rješenje za dati problem bazira se da MASTER ima osam, trenutno, aktivnih veza sa SLAVE jedinicama, kojima je neophodno upravljati, dok sve ostale jedinice su u pripravnosti tj. očekuju iniciranje veze od MASTERA



Sl. 5. *Metoda vremenskog multipleksiranja.*

Prilikom inicijalizacije elektronskog sistema potrebno je da MASTER mapira adrese svih SLAVE jedinica i da su mu te adrese konstantno dostupne u toku rada. Princip rada zasniva se da ukoliko je potrebno poslati set komandi ka npr. SLAVE1, MASTER najprije pronalazi njegovu adresu, uspostavlja vezu i šalje set komandi. Nakon toga potrebno je da sačeka odgovarajući vremenski interval kako bi dobio potvrdu o prijemu komandi ili eventualno nekih drugih podataka nakon čega se veza prekida. Problem nastaje ukoliko sa odgovarajućim SLAVE-om je potrebna neprekidna komunikacija. U tom slučaju njemu se postavlja prioritet i jedna veza je definisana samo za njega. Ukoliko u mikroračunarskom sistemu imamo više od 8 prioritetnih SLAVE kontrolera pored ovakvog rješenja moramo koristiti i metodu binarnog stabla.

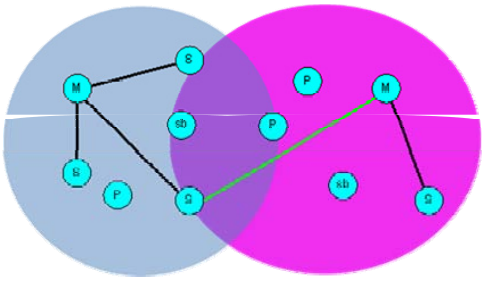
Prednost ovog rješenja jeste da se ne moraju dodavati nikakve nove jedinice kako bi se uspostavla komunikacija u sistemu. Kako se veza MASTER-SLAVE ne održava konstantno, to je i potrošnja sistema znatno smanjena. Jedina mana jeste što na ovakav način MASTER može da mapira do 255 SLAVE jedinica [4].

Ovakva arhitektura sistema je idealna za sisteme koji vrše regulaciju, upravljanje, sporo promjenjljivih veličina, tj. gdje je protok podataka MASTER-SLAVE sporadična.

5. BLUETOOTH, OSNOVNE KARAKTERISTIKE

Bluetooth je namjenjen za bežični prenos podataka i govora, kratkog dometa, namjenjen za za malu potrošnju i jeftine bežične komunikacije koje se bazraju na radio tehnologiji. Bluetooth specifikacija je IEEE standard pod nazivom 802.15 WPAN (*Wireless Personal Area Networking*).[1]

Bluetooth specifikacijom se definiše kako se *bluetooth* uređaji, prilikom komunikacije, grupišu. Jedna BT-WAP (*Bluetooth Wireless Personal Area Network*) se sastoji od *piconet*-a. Svaki *piconet* predstavlja grupa (*cluster*) od osam Bluetooth uređaja. Jedan od uređaja je *master* (glavni) a svi ostali su *slave* (podčinjeni). Dva *piconet*-a se mogu međusobno povezati preko zajedničkog Bluetooth uređaja (*gateway, bridge*) formirajući *scatternet.* Međusobno povezani *piconet*-i u okviru *scatternet*-a formiraju *backbone* (kičmu) za MANET (*Mobile Area Network*). MANET omogućava da dva uređaja koja su međusobno van dometa pokrivanja mogu razmjenjivati podatke posredstvom nekoliko skokova (*hops*) u *scatternet*-u. [1]



Sl. 6. *Izgled mrežne topologije. M-Master, S-Slave.*

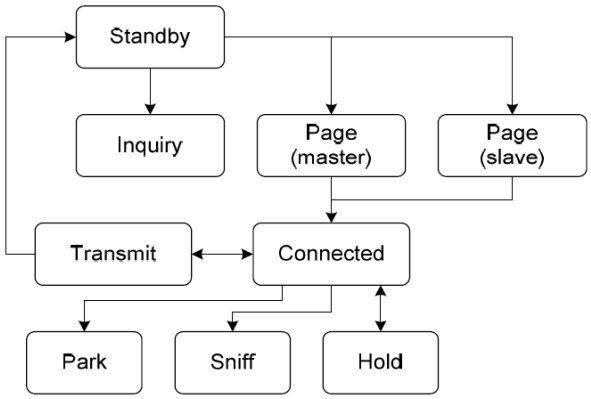
Uređaji u jednom piconet-u dijele zajednički komunikacioni kanal podataka. Ukupni kapacitet tog kanala iznosi 1 Mb/s. Zaglavlja i komunikacione informacije zauzimaju oko 20% ovog kapaciteta. Opseg frekvencija koje bluetooth pokriva kreće se između 2400 do 2483.5 MHz sa 79 1-MHz radio kanala. Svaki kanal je podijeljen na vremenske slotove u trajanju od 625μs i prilikom komunikacije *Master* uređaj šalje informacije u parnim vremenskim intervalima a *Slave* u neparnim.

Bluetooth komunikacija između dva uređaja može da bude sinhrona (SCO, *Synchronous Connection Oriented*) i asinhrona (ACL *Asynchronous Connectionless*)

Bluetooth-ov primopredajnik je uređaj koji koristi FHSS (*Frequency –Happing Spread-Spectrum*). Po međunarodnim regulativama maksimalna predajna snaga je 1W, pri čemu se samo 75 od 79 kanala koristi na pseudoslučajan način za potrebe FHSS-a Uređaj ne može da radi na dadom kanalu duže od 0.4s u okviru perioda od 30s. Ovakva ograničenja su uvedena sa ciljem da se minimizuje interferencije u ISM bendu koji se takođe koriste od strane dugih uređaja. sa ostalim.[3]

Bluetooth uređaj može biti u jednom od sledećih stanja, slika 7. , :

- pasivno (*standbye*)
- pretraživanje (*inquiry*)
- stranično (*page*)
- povezan (*connected*)
- predaja (*transmit*)
- zamrznuto (*hold*)
- parkirano (*park*)
- njuškanje (*sniff*)



Sl. 7. *Stanja bluetooth uređaja prilikom povezivanja u piconet-u*

Uređaj je u *standby* režimu rada kada je uključen na napajanje, ali još nije priključen *piconet*-u. Nakon toga prelazi u stanje *inquiry* kako bi pronašao druge uređaje sa kojima bi mogao da se poveže. Nakon uspiješne konekcije, nakon što su se odredili načini komunikacije ko je *master* a ko *slave*, započinje sa slanjem podataka. Stanje *sniff* karakteriše se malom potrošnjom uređaja, tj. to je stanje u kome je *slave* neaktivan unaprijed određeni broj vremenskih slotova. Nakon toga se uređaj vraća u neaktivno stanje sve dok ne pristigne naredni naznačeni *sniff* vremenski slot. Stanje *hold* je drugo *low-power* stanje u kojem *slave* nije aktivan određeni vremenski period, s´ tim da nema nikakvog prenosa podataka. Ukoliko *master* nema više nikakvih

informacija da šalje *slave*-u, on mu može narediti da uđe u stanje *park*. Ukoliko *slave* uđe u stanje *park*, odriče se svoje aktivne adrese u piconet-u, a ta adresa se dodjeljuje drugom *slave*-u kog *master* reaktivira iz stanja *park*.

6. POREĐENJE BLUETOOTH SA OSTALIM WIRELESS TEHNOLOGIJAMA

Pored Bluetooth-a, za komunikaciju između uređaja, danas se koriste protokoli kao što su WiFi (*Wireless Fidelity*, 802.11), HomeRF i IrDA.

802.11 standard ima bolje karakteristike u pogledu brzine prenosa signala od *bluetooth* tehnologije, jer danas dostiže 5Mb/s, što je u principu i teoretski maksimum zbog širine kanala koji koristi u proširenom spektru sa tehnikom frekvencijskog skakanja. Međutim, ya rayliku od *bluetooth*-a 802.11 ne podržava klasičen prenos govornog signal, već samo VoIP.

IrDA je jeftina tehnologija. Međutim problem je što zahtjeva neophodnu optičku vidljivost, i komunikacija je moguća samo point to point.

7. ZAKLJUČAK

U ovom radu predstavljen je elektronski sistema zasnovan na *bluetooth* komunikaciji. Data su različita arhitekturna rješenja sistema, predočeni problemi koji mogu nastati i načini njihovih rješavanja. Navedene su osnovne karakteristike *bluetooth* komunikacije, navedene prednosti u odnosu na slične standarde koje rade u istom frekvencijskom opsegu. Uzevši u obzir sve karakteristike *bluetooth*-a nameće se zaključak da ovaj standard predstavlja idealno rješenje za sinhronzovanje elektronskih sistema, jer ispunjava sve neophodne uslove za njegovo funkcionisanje.

LITERATURA

[1] Franklin C., *How Bluetooth works*, 2003

[2] ***www.bluetooth.org***

[3] ***www.bluetooth.com***

[4] www.atmel.com

Abstract – Modern electronics systems have to deal with very complicated problems. That system can be hardly made as one compact hardware entity, so it is decomposed on smaller logical parts which linked together create system of control. The way of connecting those parts have great impact on the speed, reliability of electronic system. Therefore, it is necessary to choose the right communication standard. This paper introduces the differet architecture of electronics systems wich are based on bluetooth standard of communications. It points out benefites of this type of communications, betwen modules in electronics system, compared to other conventional communicaton standards which are working in same frequency spectrum.

ELECTRONICS SYSTEMS BASED ON BLUETOOTH COMUNICATION

D. Gurović

ODREĐIVANJE PARAMETARA EKVIVALENTNE ŠEME I VEKTORSKA KONTROLA ASINHRONE KAVEZNE MAŠINE

Milorad Burmazović, Dejan Reljić, Veran Vasić, *Fakultet tehničkih nauka, Novi Sad, miloradburmazovic@yahoo.com*
Petar Jerkan, *ATB Sever, Subotica, Petar.Jerkan@yu.atb-motors.com*

Sadržaj – *U radu je prikazan postupak određivanja parametara ekvivalentne šeme trofaznog kaveznog asinhronog motora kao i praktična realizacija vektorske regulacije. U cilju određivanja parametara ekvivalentne šeme asinhrone mašine obavljen je ogled praznog hoda i kratkog spoja pri nazivnoj učestanosti napajanja, kao i ogled određivanja reaktanse praznog statora. Podaci dobijeni na ovaj način su iskorišćeni prilikom eksperimentalne realizacije vektorske kontrole. Na osnovu matematičkog modela vektorski regulisanog asinhronog motora izvršena je simulacija rada pogona uz pomoć programskog paketa Matlab. Razmotren je slučaj razdešenih parametara. Sproveden je i eksperimentalni ogled. Upravljački algoritam je realizovan pomoću razvojnog sistema ACE 1104, odnosno upravljačke kartice DS1104.*

1. UVOD

Potreba za automatizacijom proizvodnih procesa i povećanjem njihove efikasnosti i pouzdanosti je dovela do intenzivnog razvoja i usavršavanja oblasti električnih pogona. U prošlosti su pogoni visokih performansi koristili komutatorske mašine jednosmerne. Posebno je bila rasprostranjena primena komutatorskih mašine sa nezavisnom pobudom, obzirom da se njihov fluks i moment mogu jednostavno regulisati promenom pobudne struje i struje kola rotora. Ipak, jednosmerne mašine imaju izvesne nedostatke koji se pre svega odnose na postojanje komutatora i četkica, pa zahtevaju periodično održavanje. Problemi se mogu prevazići upotrebom mašina naizmenične struje koje imaju jednostavniju i pouzdaniju strukturu (manju potrebu za održavanjem) i veću ekonomičnost. Robusne su i otporne na velika preopterećenja i manjih su geometrijskih dimenzija u poređenju sa mašinama jednosmerne struje.

Među različitim elektromotornim pogonima naizmeničnih mašina, izdvojili su se oni koji koriste asinhrone mašine i to sa kaveznim rotorom. Takvi pogoni imaju posebne prednosti, pre svega ekonomske. Kavezna asinhrona mašina je jednostavne konstrukcije, robusna je i pouzdana i jedna je od najjeftinijih mašina u oblasti kako malih tako i velikih snaga.

Praktična realizacija složenijh algoritama upravljanja asinhronom kaveznom mašinom, kao što je i vektorska regulacija, omogućena je razvojem brzih digitalnih signalnih procesora [1]. Vektorska regulacija obezbeđuje: nezavisnu kontrolu momenta i fluksa, pun moment motora pri malim brzinama, rad u sva četiri kvadranta, veću efikasnost u širem opsegu brzina i kvalitetniju dinamiku u prelaznim režimima rada [1]. Da bi se u pogonu mogla realizovati vektorska regulacija, potrebno je pre svega utvrditi položaj prostornog fazora fluksnog obuhvata. U početku primene vektorske regulacije položaj polja se određivao na osnovu merenja fluksa u mašini pomoću senzora magnetne indukcije. Takva regulacija je poznata kao direktna vektorska regulacija [1]. Danas se ne primenjuje zbog problema koje sa sobom nose senzori magnetne indukcije. Zbog toga je razvijena indirektna vektorska regulacija kod koje se položaj prostornog fazora fluksnog obuhvata određuje merenjem položaja rotora pomoću davača i obračunavanjem efekta klizanja [1].

Za nezavisno upravljanje fluksom i momentom asinhronog motora potrebno je poznavati parametre mašine. Induktivnosti statorskog i rotorskog kola zavise od nivoa zasićenja u mašini, dok se otpornosti namota menjaju sa temperaturom i učestanošću. Odlučujući uticaj na karakteristike i kvalitet pogona sa indirektnom vektorskom kontrolom ima rotorska vremenska konstanta. Iz tog razloga će se prvo odrediti parametri ekvivalentne šeme mašine, a nakon toga realizovati vektorska kontrola.

2. ODREĐIVANJE PARAMETARA EKVIVALENTNE ŠEME

Merenje otpora namota statora je vršeno UI metodom [2]. Izmerena srednja vrednost otpora namota statora je 41,98 Ω. Merenje je izvršeno na temperaturi ambijenta od 19 °C. Otpornosti namota statora preračunate na temperaturu ambijenta od 25 °C iznose R_s=43,05 Ω.

Merenje reaktanse praznog statora se vrši na motoru iz koga je prethodno uklonjen rotor [3]. Stator se priključi na mrežu, a napon napajanja se podesi tako da vrednost struje statora bude nominalna. Dovedeni napon se raspodeljuje na: indukovani napon usled rasipanja, napon indukovan usled obrtnog fluksa koji se zatvara kroz šupljinu statora i napon na termogenoj otpornosti statora. Izmerene vrednosti napona i struje su U=45,76 V i I=0,78 A. Reaktansa rasipanja statora se izračunava na sledeći način [3]:

$$X_{ys} = \sqrt{\left(\frac{U}{I}\right)^2 - R_s^2} - X_o, \qquad (1)$$

gde je:

$$X_o = \frac{125}{6\pi}\left(\frac{N\cdot k}{100}\right)^2 \cdot \frac{f}{50}\cdot\frac{1}{2p}\left[l_{Fe} + \frac{\tau}{6}\right], \qquad (2)$$

N – broj navojaka po fazi statora (992),
k – navojni sačinilac statorskog namota (0,96593),
f – učestanost napona napajanja (50 Hz),
p – broj pari polova (2),
l_{Fe} – dužina paketa limova statora (0,035 m),
τ - polni korak (0,04867 m).

Izračunavanjem se dobija X_o=6,562 Ω. Otpornost namota statora u ovom ogledu je izmerena i iznosi R_s=43,45 Ω. Otpornost se promenila usled zagrevanja namota. Reaktansa rasipanja statora prema izrazu (1) je X_{ys}=32,86 Ω.

Ogled praznog hoda se vrši kako bi se odredila reaktansa magnećenja. U ogledu se napon napajanja nazivne učestanosti menja u opsegu $(1{,}15\div 0{,}75)U_n$ i pri tome se mere struje u sve tri faze i aktivna snaga. S obzirom da je motor potpuno rasterećen, brzina mu je bliska sinhronoj, a klizanje je $s\approx 0$. Struja rotora je zanemarljivo mala, tako da se grana koja modeluje rotor može zanemariti [2]. Na osnovu

Napomene:
a) Rad je proistekao iz diplomskog-master rada Milorada Burmazovića. Mentor je bio doc.dr Veran Vasić.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

izmerene aktivne snage u praznom hodu P_0 i faktora snage $\cos\varphi_0$, može se izračunati reaktivna snaga mašine:

$$Q_0 = P_0 \cdot \mathrm{tg}\varphi_0 \tag{3}$$

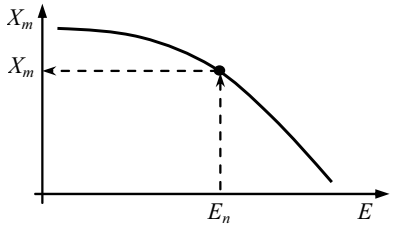
Reaktivna snaga se "troši" na reaktansi rasipanja statora i reaktansi magnećenja:

$$Q_0 \approx 3 \cdot X_{\gamma s} \cdot I_0^2 + 3 \cdot \frac{E^2}{X_m} \tag{4}$$

Vrednost indukovane elektromotorne sile na grani magnećenja se određuje iz relacije:

$$\overline{E} = \overline{U}_0 - (R_s + jX_{\gamma s}) \cdot \overline{I}_0 \tag{5}$$

Na osnovu tako dobijenih rezultata nacrtan je grafik $X_m=f(E)$, koji je prikazan na Sl. 1.



Sl. 1. *Grafik zavisnosti $X_m=f(E)$*

Sa grafika je moguće odrediti reaktansu magnećenja X_m koja se ima u nominalnom režimu rada mašine. Za to je prethodno potrebno izračunati elektromotornu silu u nominalnom režimu rada, a na osnovu nominalnih podataka sa natpisne pločice mašine. Nominalna elektromotorna sila ima vrednost 178 V, a sa grafika se očitava reaktansa magnećenja: X_m=391 Ω. Nominalna struja magnećenja je:

$$I_{mn} = \frac{E_n}{X_m} = \frac{178}{392} = 0{,}454 \text{ A} \tag{6}$$

Ogled kratkog spoja se vrši radi određivanja preostalih parametara ekvivalentne šeme (reaktanse rasipanja rotora svedene na stranu statora i termogene otpornosti rotora svedene na stranu statora). Rotor asinhronog motora je mehanički ukočen [2]. Napon napajanja motora je podešen na vrednost pri kojoj se ima nazivna struja. Pri tome se vrši merenje napona, struje i aktivne snage. Srednje vrednosti izmerenih faznih napona i struja su: U_k=73,5 V i I_k=0,79 A. Izmerena aktivna snaga je P_k=128,48 W. S obzirom da kroz namotaj protiče nominalna struja, isti će se zagrejati. Zbog toga se nakon obavljenog ogleda meri i otpornost namota statora, koja u ovom slučaju iznosi R_s=43,73 Ω. Aktivna energija koju motor povlači iz mreže se "troši" na otpornostima namotaja statora i rotora (gubici u gvožđu se zanemaruju):

$$P_k \approx 3 \cdot (R_s + R_r) \cdot I_k^2 \tag{7}$$

Iz prethodne jednačine se izračunava termogena otpornost namota rotora svedena na stranu statora. Ona iznosi R_r=24,89 Ω. Reaktivna snaga motora u kratkom spoju je:

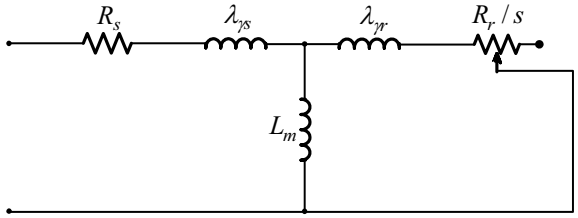
$$Q_k = P_k \cdot \mathrm{tg}\varphi_k = 117{,}63 \text{ VAr} \tag{8}$$

Ova reaktivna energija se raspodeljuje na reaktansama rasipanja statora i rotora (zanemarena je grana magnećenja).

$$Q_k = 3 \cdot (X_{\gamma s} + X_{\gamma r}) \cdot I_k^2 \tag{9}$$

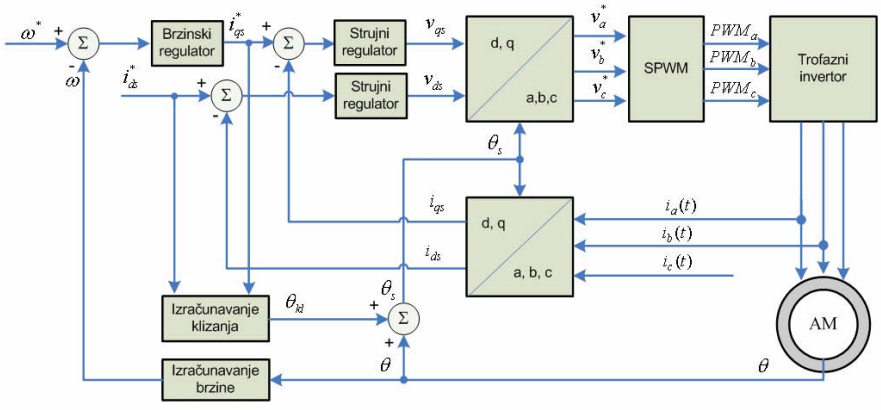
Na osnovu izraza (8) i (9) se izračunava nepoznata reaktansa rasipanja rotora svedena na stranu statora: $X_{\gamma r}$=30 Ω.

Ekvivalentna šema asinhrone mašine [4] je prikazana na Sl. 2, a parametri ove šeme su: R_s=43,05 Ω, $\lambda_{\gamma s}$=104,6 mH, L_m=1244,6 mH, $\lambda_{\gamma r}$=95,5 mH i R_r=24,89 Ω.



Sl. 2. *Ekvivalentna šema asinhrone mašine*

3. VEKTORSKA KONTROLA



Sl. 3. *Blok šema vektorski regulisane asinhrone mašine*

Osnovni zahtev koji se postavlja vektorskom upravljanju jeste nezavisno upravljanje fluksom i momentom koji mašina razvija [1], kako bi regulacija pogona sa asinhronom mašinom postala ekvivalentna regulaciji pogona jednosmerne mašine sa nezavisnom pobudom. Nezavisno upravljanje fluksom i momentom asinhronog motora se može ostvariti kontrolom amplitude vektora statorske struje i orijentacije u odnosu na prostorni vektor fluksa mašine. Orijentacija može biti u odnosu na fluks statora, fluks rotora i fluks magnećenja. Orijentacija u odnosu na fluks rotora ima najmanji stepen

složenosti u praktičnoj realizaciji i o njoj će ovde biti reči. Blok šema indirektne vektorske regulacije je prikazana na Sl. 3. Posmatra se pogon regulisan po brzini. Ako se izabere koordinatni sistem vezan za prostorni fazor fluksnog obuhvata rotora tako da je poprečna komponenta prostornog fazora fluksnog obuhvata rotora jednaka nuli (Ψ_{rq}=0), tada izraz za elektromagnetni momenat motora glasi [1]:

$$m_e = \frac{3}{2} \cdot p \cdot \frac{L_m}{L_r} \cdot \psi_{rd} \cdot i_{sq} \tag{10}$$

pri čemu direktna komponenta rotorskog fluksa u stacionarnom stanju ima oblik [1]:

$$\psi_{rd} = L_m \cdot i_{sd} \tag{11}$$

Na ovaj način je postignuto upravljanje fluksom uz pomoć struje statora po d osi (i_{sd}) i elektromagnetnim momentom koristeći struju statora po q osi (i_{sq}). Kod vektorske regulacije motor se najčešće napaja iz strujno regulisanog naponskog izvora. Dakle, kontrolom amplitude i učestanosti vektora statorske struje i njegove orjentacije u odnosu na vektor rotorskog fluksa ostvaruje se raspregnuto upravljanje fluksom i momentom asinhronog motora.

Vektorsko upravljanje zahteva poznavanje položaja vektora fluksa rotora. Kod indirektne vektorske regulacije (Sl. 3) položaj prostornog fazora fluksnog obuhvata rotora se određuje merenjem položaja rotora pomoću davača pozicije ili brzine i obračunavanjem efekta klizanja [1]. Vrednost klizanja se izračunava na sledeći način:

$$\omega_{kl} = \frac{1}{T_r} \cdot \frac{i_{sq}}{i_{sd}} \tag{12}$$

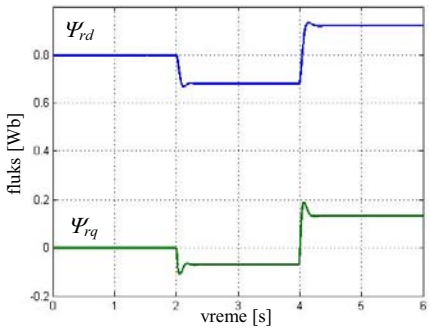
Položaj vektora fluksa rotora je:

$$\theta_s = \theta + \int \omega_{kl} \cdot dt \tag{13}$$

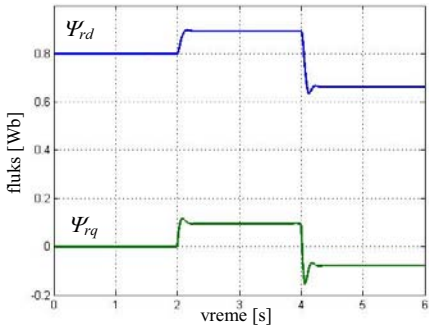
U nastavku će biti analiziran slučaj uticaja pogrešno podešene vrednosti rotorske vremenske konstante T_r na rad pogona koji je indirektno vektorski upravljan.

Za regulisani pogon sa indirektnom vektorskom kontrolom (Sl. 3) napravljena je računarska simulacija u programskom paketu *Matlab/Simulink* kojom je ilustrovan gubitak vektorske kontrole u pogonu usled pogrešno podešene vrednosti otpornosti rotora R_r i međuinduktivnosti L_m u upravljačkom sistemu. Na Sl. 4 su prikazani fluksevi asinhrone mašine koji se imaju prilikom promene R_r u upravljačkom sistemu. Pretpostavljeno je da je do trenutka t=2 s otpornost rotora u modelu korektno podešena, u trenutku t=2 s otpornost rotora u modelu se povećala za 25 %, a u trenutku t=4 s otpornost rotora je smanjena za 25 % u odnosu na tačnu vrednost. Na osnovu rezultata sa Sl. 4 može se zaključiti da je orijentacija sinhrono rotirajućeg koordinatnog sistema pogrešna. Gubitak vektorske kontrole zbog promene R_r se ogleda u pojavi fluksa rotora Ψ_{rq}.

Na Sl. 5 su prikazani fluksevi asinhrone mašine koji se imaju prilikom promene L_m u upravljačkom sistemu. Pretpostavljeno je da je do trenutka t=2 s međuinduktivnost u modelu korektno podešena, u trenutku t=2 s međuinduktivnost se povećala za 25 %, a u trenutku t=4 s međuinduktivnost je smanjena za 25 % u odnosu na tačnu vrednost. Na osnovu rezultata sa Sl. 5 može se zaključiti da pogon nema vektorsku orijentaciju jer se pojavila poprečna komponenta fluksa rotora Ψ_{rq}. Amplituda fluksa rotora u podužnoj osi Ψ_{rd} odstupa od zadate vrednosti. Tokom obe simulacije motor je bio nominalno opterećen, a brzina obrtanja nominalna.



Sl. 4. *Fluksevi asinhrone mašine pri promeni R_r*

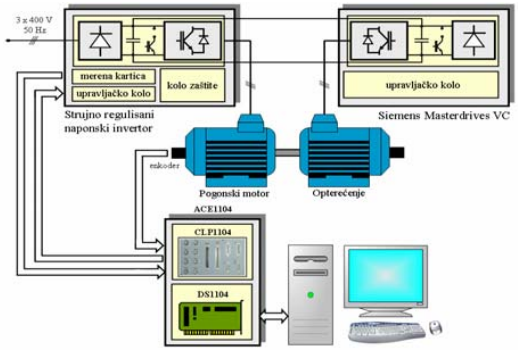


Sl. 5. *Fluksevi asinhrone mašine pri promeni L_m*

Na osnovu izloženih rezultata simulacije može se zaključiti da je za kvalitetno izvođenje vektorske regulacije potrebno što tačnije poznavanje parametara mašine, pre svega rotorske vremenske konstante T_r.

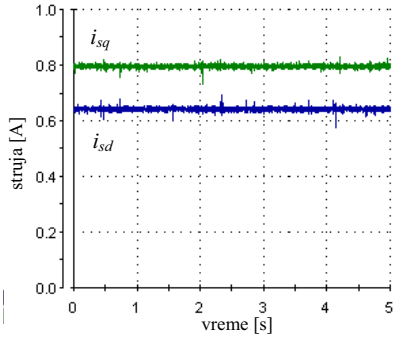
4. EKSPERIMENTALNA VERIFIKACIJA

Eksperimenti su vršeni na grupi koju čine dva identična trofazna četvoropolna asinhrona motora. Jedan asinhroni motor (snage 250 W) predstavlja pogonsku mašinu, a drugi asinhroni motor radi u momentnom režimu i predstavlja opterećenje pogonskoj mašini. Parametri pogonskog motora su eksperimentalno određeni i dati su u drugom delu ovog rada. Za napajanje pogonskog asinhronog motora je korišćen trofazni naponski invertor. Upravljački algoritmi su realizovani pomoću razvojnog sistema ACE 1104, odnosno pomoću upravljačke kartice DS1104, koja podržava programiranje procesora na programskom jeziku višeg nivoa. U konkretnoj realizaciji kompletan softver je realizovan u programskom paketu *Matlab/Simulink* i programskom okruženju razvojnog sistema ACE1104 za razvoj aplikacija u realnom vremenu.

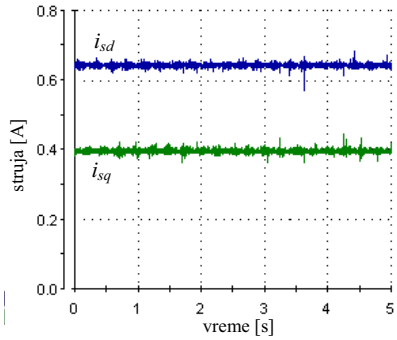


Sl. 6. *Blok šema realizovanog pogona*

Ogled je izvršen na vektorskom pogonu kome je zadata nulta referenca brzine. Na taj način je ostvareno da je moment usled mehaničkih gubitaka ravan nuli. Ovo je bitno zbog poređenja vrednosti zadatog momenta opterećenja i estimiranog elektromagnetnog momenta vektorski regulisanog motora. Na narednim graficima su prikazane komponente struje statora i_{sd} i i_{sq} prilikom opterećenja motora sa dve različite vrednosti momenta opterećenja. Moment opterećenja u prvom slučaju je 1,6 Nm (Sl. 7), a u drugom dva puta manji (Sl. 8). Kako se stuja i_{sq} u ova dva slučaja takođe promenila dva puta, pri čemu se i_{sd} nije promenila, može se izvesti zaključak da je vremenska konstanta rotora korektno određena.



Sl. 7. *Struje i_{sd} i i_{sq} pri momentu opterećenja od 1,6 Nm*



Sl. 8. *Struje i_{sd} i i_{sq} pri momentu opterećenja od 0,8 Nm*

Na osnovu grafika sa Sl. 7 i Sl. 8 i izraza (10), može se izračunati elektromagnetni moment vektorski regulisanog motora. Za grafik sa Sl. 7 elektromagnetni moment motora iznosi m_e=1,76 Nm, a za grafik sa Sl. 8 elektromagnetni moment motora iznosi m_e=0,88 Nm. Razlika zadatog momenta opterećenja i estimiranog momenta motora je približno 10 % u oba slučaja. Uzrok ovog neslaganja treba tražiti u parametrima mašine: L_m i L_r (odnosno λ_{yr}). Naime induktivnost magnećenja je određena iz ogleda praznog hoda pri čemu je zanemarena grana rotora. Zbog postojanja mehaničkih gubitaka u realnom praznom hodu i kroz namotaj rotora će proticati struja, te je stvarna induktivnost magnećenja nešto manja od prethodno određene vrednosti. Reaktansa rasipanja rotora je određena iz ogleda kratkog spoja. Međutim, u stvarnosti će ova reaktansa biti nešto veća, a samim tim i induktivnost rotorskog namotaja će biti veća.

Takođe, treba napomenuti da je i sam matematički model električne mašine približna predstava pojava koje se dešavaju u električnoj mašini. Model je izveden pod određenim pretpostavkama [4] koje u stvarnosti važe u nekoj meri, ali ne u potpunosti.

5. ZAKLJUČAK

U radu je predstavljen jedan način određivanja parametara ekvivalentne šeme trofaznog kaveznog asinhronog motora. U tu svrhu je izveden ogled praznog hoda i kratkog spoja, ali i ogled za određivanje reaktanse rasipanja statora.

Realizovana je vektorska kontrola pogona, a parametri ekvivalentne šeme, koji su prethodno određeni, su provereni u datom pogonu. Upravljački algoritam je realizovan pomoću razvojnog sistema ACE 1104, odnosno pomoću savremene digitalne upravljačke kartice DS1104, koja podržava programiranje procesora na programskom jeziku višeg nivoa.

Na osnovu dobijenih rezultata eksperimenta je utvrđeno da je vremenska konstanta rotora ispravno određena. Postojanje malih neslaganja između stvarnog momenta opterećenja i estimiranog elektromagnetnog momenta motora je posledica izvesnog odstupanja izračunate induktivnosti magnećenja i njene stvarne vrednosti. Takođe, i sam matematički model električne mašine je približan.

Dalja istraživanja treba usmeriti na *on-line* identifikaciju vremenske konstante rotora, jer je ona podložna promenama usled zagrevanja, a ima odlučujući uticaj na karakteristike i kvalitet pogona sa indirektnom vektorskom kontrolom.

LITERATURA

[1] V. Vučković, *Električni pogoni*, Beograd: Akademska misao, 2002.

[2] M. Petrović, *Ispitivanje električnih mašina*, Beograd: Naučna knjiga, 1988.

[3] G. Rašković, *Određivanje zagrejanja i prevalnog momenta trofaznog asinhronog kaveznog motora*, Novi Sad: Fakultet tehničkih nauka, diplomski rad, 1989.

[4] V. Vučković, *Opšta teorija električnih mašina*, Beograd: Nauka, 1992.

Abstract – This paper presents a procedure for equivalent circuit parameters determination of induction squirrel cage motor drive. For that purpose no load test and locked rotor test are performed, as well as test for stator leakage reactance determination. Detuning of induction motor parameters cause errors in flux orientation. This is proven by mathematical simulation. Results of parameters determination are verified by a laboratory prototype of speed vector controlled induction motor drive. The control structure is programmed as a *Simulink* model and it is implemented by Real – Time Interface on DS1104 Controller Board.

EQUIVALENT CIRCUIT PARAMETERS DETERMINATION AND VECTOR CONTROL OF INDUCTION SQUIRREL CAGE MOTOR DRIVE

Milorad Burmazović, Dejan Reljić, Veran Vasić, Petar Jerkan

UTICAJ GREŠKE PARAMETRA VREMENSKE KONSTANTE ROTORA NA RAD INDIREKTNE VEKTORSKE KONTROLE

INFLUENCE OF ROTOR TIME CONSTANT ERROR ON IFOC CONTROL STRUCTURE

Janoš Timer, Evgenije Adžić, Vlado Porobić, Darko Marčetić *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U ovom radu je prikazan uticaj pogrešno određene vremenske konstante rotora kaveznog asinhronog motora na upravljanje asinhronim motorom algoritmom indirektne vektorske kontrole (IFOC – indirect field oriented control). Simulacija i eksperimentalni rezultati su pokazali značajnu osetljivost pogona na neslaganje parametara, pogotovo kod velikih opterećenja pogona. Na osnovu toga je zaključeno da je u slučaju velikih promena rotorske vremenske konstante potrebno implementirati mehanizam njene estimacije tokom celokopnog trajanja rada pogona.*

Abstract – *The rotor time constant detuning sensitivity of IFOC drive is investigated in this paper. The simulation, as well as, experimental results show significant influence of parameter mismatch, especially for high load values. Therefore, if significant rotor time constant drift is expected, an on-line estimation mechanism is required*

Ključne reči: *Vremenska konstanta rotora, IFOC pogon, DSP*

1. UVOD

Potreba za automatizacijom proizvodnih procesa i povećanjem njihove efikasnosti i pouzdanosti je dovela do usavršavanja oblasti električnih pogona. Ranije su se u pogonima visokih performansi koristile komutatorske jednosmerne mašine. Izrazito je bila rasprostranjena primena jednosmernih mašina sa nezavisnom pobudom, s obzirom da se njihov fluks i električni momenat mogu jednostavno i međusobno nezavisno regulisati promenom pobudne struje i struje kola rotora. Zbog izvesnih nedostataka jednosmernih mašina koji se pre svega odnose na postojanje komutatora i četkica, zahteva se često periodično održavanje. Ovaj problem se prevazilazi upotrebom mašina naizmenične struje koje imaju jednostavniju i pouzdaniju strukturu (manju potrebu za održavanjem) i veću ekonomičnost. Robusne su i otporne na velika preopterećenja i manjih su geometrijskih dimenzija u poređenju sa mašinama jednosmerne struje.

NAPOMENA:

a) Ovaj rad je proistekao iz istoimenog diplomskog-master rada čiji mentor je bio dr Darko Marčetić, docent

b) Ovaj rad je prethodno publikovan na konferenciji Indel, 26-28. nov. 2008., Banja Luka.

U različitim elektromotornim pogonima, izdvojili su se naizmenični pogoni koji koriste asinhrone mašine sa kaveznim rotorom u oblasti kako malih tako i velikih snaga. Vektorska regulacija asinhronog motora se praktično vrši primenom, brzih digitalnih signalnih procesora (DSP). Ona omogućuje nezavisnu kontrolu električnog momenta i fluksa, pun elektromehanički momenat motora pri malim brzinama, rad u sva četiri kvadranta, veću efikasnost u širem opsegu brzina i kvalitetniju dinamiku u prelaznim režimima rada [1]. Za realizaciju vektorske regulacije u pogonu, potrebno je pre svega utvrditi položaj prostornog vektora rotorskog fluksnog obuhvata. Indirektna vektorska kontrola pogona (IFOC) je jedan od načina za određivanje prostornog fazora fluksnog obuhvata merenjem položaja rotora pomoću davača pozicije rotora i obračunavanjem efekta klizanja [1].

Kvalitet IFOC pogona zavisi od tačnosti postupka određivanja parametra vremenske konstante rotora. Iz tog razloga je potrebno što tačnije odrediti parametre ekvivalentne šeme mašine [2].

2. MODEL ASINHRONOG MOTORA SA DVE PROMENLJIVE STANJA

Asinhroni motor je relativno proste konstrukcije ali iziskuje složeno upravljanje. Sam model asinhrone mašine je moguće izgraditi na osnovu različitih promenjivih stanja. Kada pogon poseduje strujno regulisani naponski invertor (CRVSI-current regulated voltage source inverter) pogodno je za promenjive stanja odabrati upravo vektor struje statora i vektor fluksa rotora. Jednačine naponskog balansa namotaja statora i rotora, kao i jednačine fluksnih obuhvata, izražene preko kompleksnih vektora napona, struja i flukseva su [3]:

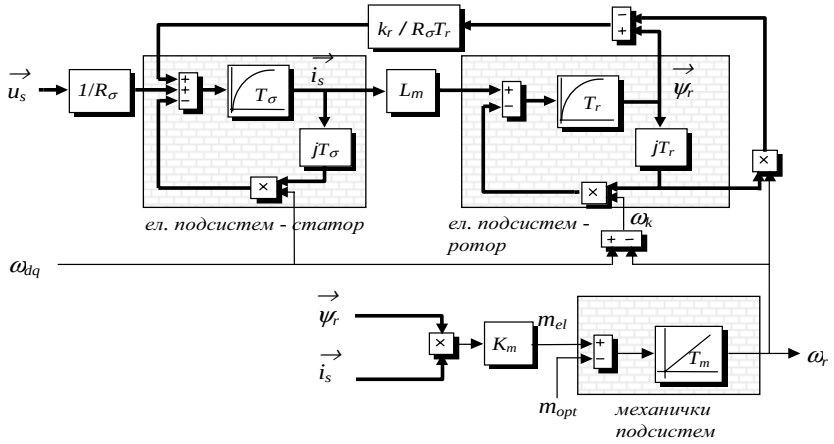
$$\begin{aligned} \vec{\mathbf{u}}_s &= R_s \vec{\mathbf{i}}_s + \frac{d\vec{\psi}_s}{dt} + j\omega_{dq}\vec{\psi}_s \\ 0 &= R_r \vec{\mathbf{i}}_r + \frac{d\vec{\psi}_r}{dt} + j(\omega_{dq} - \omega_r)\vec{\psi}_r \end{aligned} \tag{1}$$

$$\begin{aligned} \vec{\psi}_s &= L_s \vec{\boldsymbol{i}}_s + L_m \vec{\boldsymbol{i}}_r \\ \vec{\psi}_r &= L_m \vec{\boldsymbol{i}}_s + L_r \vec{\boldsymbol{i}}_r \end{aligned} \tag{2}$$

gde su:

$$\vec{\mathbf{u}}_s = \begin{bmatrix} u_{ds} \\ u_{qs} \end{bmatrix},\ \vec{\mathbf{i}}_s = \begin{bmatrix} i_{ds} \\ i_{qs} \end{bmatrix} \text{ i } \vec{\psi}_r = \begin{bmatrix} \Psi_{dr} \\ \Psi_{qr} \end{bmatrix}$$

vektori napona, struje statora i fluksa rotora, ali u ***dq*** koordinatnom sistemu koji sinhrono rotira sa poljem ugaone brzinom ω_{dq}. Diferencijalna jednačina fluksa rotora je (3).



Sl. 1. Kompleksni model mašine sa vektorima struje statora i fluksa rotora kao promenjivima stanja

$$\frac{d\vec{\psi}_r}{dt} = -R_r\vec{i}_r - j(\omega_{dq} - \omega_r)\vec{\psi}_r \tag{3}$$

Kompleksne jednačine mašine su:

- *Statorske jednačine modela:*

$$T_\sigma \frac{d\vec{i}_s}{dt} + \vec{i}_s = -j\omega_{dq}T_\sigma\vec{i}_s + \frac{k_r}{R_\sigma T_r}(1 - j\omega_r T_r)\vec{\psi}_r + \frac{1}{R_\sigma}\vec{u}_s \tag{4}$$

- *Rotorske jednačine modela:*

$$T_r \frac{d\vec{\psi}_r}{dt} + \vec{\psi}_r = -j(\omega_{dq} - \omega_r)T_r\vec{\psi}_r + L_m\vec{i}_s \tag{5}$$

gde su:

$$T_\sigma = \frac{L_\sigma}{R_\sigma}, L_\sigma = L_s(1 - \frac{L_s L_r}{L_m{}^2}),$$

$$R_\sigma = R_s + R_r k_r{}^2, \; k_r = \frac{L_m}{L_r}, T_r = \frac{L_r}{R_r}$$

Jednačina mehaničkog podsistema je data sa

$$T_m \frac{d\omega_r}{dt} = \frac{3}{2} p \frac{L_m}{L_r}(\vec{\psi}_r \times \vec{i}_s) - m_{opt} \tag{6}$$

u kojoj je elektromagnetni momenat prikazan kao vektorski proizvod fluksa rotora i struje statora. Blok dijagram toka kompleksnih signala ovakvog modela je prikazan na Sl.1

3. INDIREKTNA PROCENA POLOŽAJA FLUKSA ROTORA

Efikasno upravljanje asinhronim motorom se može ostvariti podešavanjem učestanosti i efektivne vrednosti napona, ili struje statora. Dodatno, izborom i odgovarajućeg faznog stava upravljačke veličine na statoru, moguće je postići iste upravljačke efekte kao kod jednosmernog motora sa nezavisnom pobudom.

Da bi se ostvarilo optimalno vektorsko upravljanje asinhronim motorom neophodno je nezavisno upravljati fluksom i ostvarenim elektromagnetnim momentom, koji mašina razvija [1]. Konture upravljanja ovim veličinama se mogu razdvojiti regulacijom amplitude magnetopobudne sile statora i njenog relativnog položaja u odnosu na vektor fluksa rotora. Magnetnopobudnom silom statora je moguće upravljati pomoću strujno regulisanog naponskog invetora (CRVSI). Eliminacijom jednačina naponskog balansa statora model električnog podsistema se redukuje i svodi samo na jednu vektorsku jednačinu (5).

U slučaju idealne strujne regulacije, u motor se utiskuje vektor struje čija je amplituda i fazni stav jednak referentnom. Ovim je moguće iz analize potpuno isključiti električni podsistem statora. Ali, da bi se raspregnutno upravljanje ostvarilo i dalje je neohpodno poznavati položaj vektora fluksa rotora.

Položaj vektora fluksa rotora se proračunava algoritmom indirektne vektorske kontrole. Po ovom algoritmu procena položaja se vrši u strujnom modelu rotorskog kola koji na osnovu vektora struje statora i mehaničkog položaja rotora simulira pojave u rotoru motora.

Ukoliko se poseduje idealan strujno regulisani naponski invertor, tada se može smatrati da su ostvarene (merene) i zadate komponente struje statora jednake. U tom slučaju se mogu koristiti referentne komponente vektora struje statora na ulazu rotorskog električnog podsistema.

Zadavanjem odgovarajućih referentnih veličina strujno regulisanog naponskog invertora moguće je u mašinu utisnuti željeni (bilo koji) kompleksni vektor ulaznih struja statora, na željenoj učestanost pobudnog polja (ω_{dq}).

Ukoliko je poznata učestanost rotora (obezbeđeno je merenje ili procena te veličine) kontrolom učestanosti pobudnog polja se praktično kontroliše učestanost klizanja ω_k.

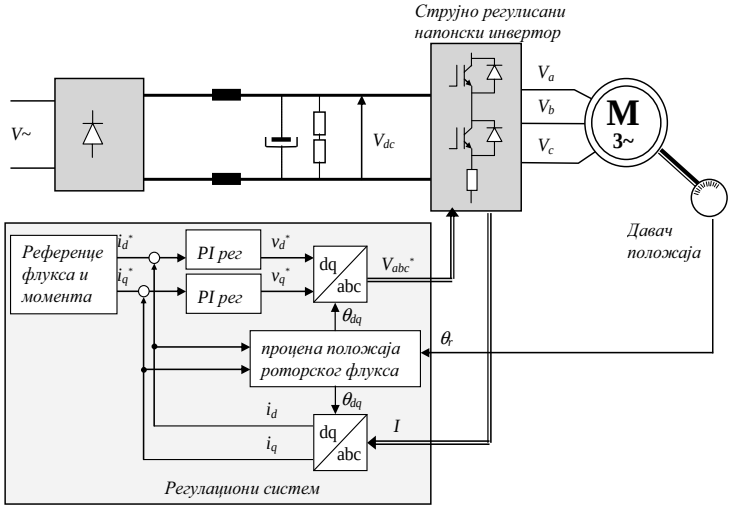
Primenom strujno regulisanog naponskog invertora model mašine jeste redukovan, ali time i dalje nije obezbeđeno raspregnuto upravljanje fluksom i elektromagnetnim momentom. Ako bi ovako ostalo, promena poprečne komponente struje statora bi i dalje izazivala i promenu fluksa rotora, a ne samo promenu momenta, koja ne bi bila ni linearna sa porastom poprečne komponente struje statora.

Strujno regulisani naponski invertor pored amplitude statorskih struja kontroliše i učestanost obrtnog polja statora ω_s. Ukoliko se poznaje električna učestanost rotora ω_r, jasno je da je moguće kontrolisati i učestanost klizanja.

Ukoliko se vektor fluksa rotora postavi tačno na d osu time se dobija raspregnuto upravljanje fluksom i momentom (vrednost i_{qs} više neće uticati na fluks rotora). To znači da ako se ψ_{qr} izjednači sa nulom, dobija se [1]:

$$\omega_k = \frac{L_m i_{qs}}{T_r \psi_{dr}} \tag{7}$$

tj. ima se potpuni nestanak fluksa rotora u q osi (8).



Sl. 2. Šematski prikaz vektorski kontrolisanog motora sa strujno regulisanim naponskim invertorom

$$T_r \frac{d\psi_{qr}}{dt} + \psi_{qr} = 0 \tag{8}$$

Time je vektor fluksa rotora na ***d*** osi:

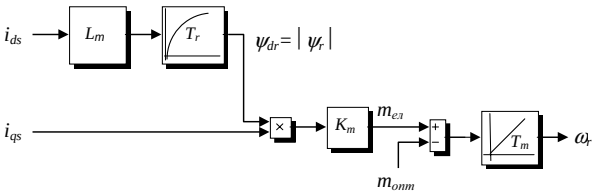
$$T_r \frac{d\psi_{dr}}{dt} + \psi_{dr} = L_m i_{ds} \tag{9}$$

što pokazuje da je na ovaj način moguće amplitudu fluksa rotora kontrolisati samo sa promenom ***d*** komponente struje statora, potpuno nezavisno od ***q*** komponente

U isto vreme, i izraz za elektromagnetni momenat se uprošćava, i za istu amplitudu fluksa se dobija linearna promena momenta sa promenom ***q*** komponente struje [1]:

$$m_{el} = K_m \psi_{dr} i_{qs} \ , K_m = \frac{3}{2} p \frac{L_m}{L_r} \tag{10}$$

Ovim su konture upravljanja fluksom i električnim momentom raspregnute i moguće je optimalno upravljati asinhronim motorom.



Sl.3. Redukovani model asinhrone mašine

4. UTICAJ POGREŠNOG PARAMETRA VREMENSKE KONSTANTE ROTORA NA INDIREKTNO VEKTORSKO UPRAVLJANJE

Porebno je da DSP sve relevantne veličine u pogonu računa u realnom vremenu (model prikazan na Sl.2.). Ukoliko su ovi proračuni tačni, položaj vektora rotorskog fluksa se tačno procenjuje i ostvareno je nezavisno upravljanje fluksom i momentom motora.

U prikazanom modelu se koristi parametar vremenske konstante rotora T_r^* na čiju grešku je rad modela veoma osetljiv. Ukoliko parametar T_r^* koji se koristi u modelu nije u skladu sa stvarnom vremenskom konstantnom rotora $T_r = L_r / R_r$ dolazi do greške u proceni položaja rotorskog fluksa. Tada konture upravljanja fluksom i momentom nisu više raspregnute.

Na osnovu matematičkog modela vektorski regulisanog kaveznog asinhronog motora pomoću simulacije rada pogona uz pomoć programskog paketa *Matlab* se uočava uticaj greške pogrešno određene vremenske konstante rotora na rad IFOC pogona. Indirektna procena vektora rotorskog fluksa u pogonu sa davačem pozicije zavisi od obračunavanja klizanja. Pri tome direktna komponenta rotorskog fluksa u stacionarnom stanju ima oblik [1]:

$$\psi_{dr} = L_m \cdot i_{ds} \tag{11}$$

pa se klizanje se može predstaviti kao

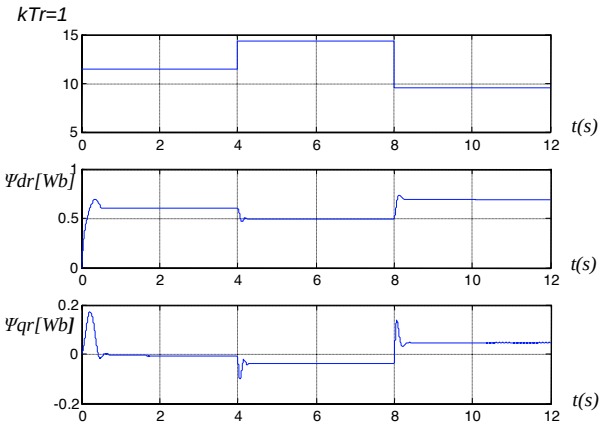
$$\omega_k = \frac{1}{T_r} \cdot \frac{i_{qs}}{i_{ds}} \tag{12}$$

U (12) i_{qs} i i_{ds} su upravljive ulazne veličine klizanja.

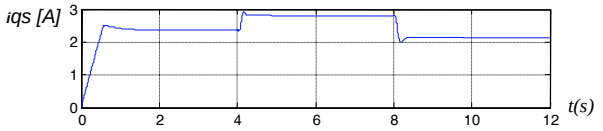
Za indirektnu vektorsku kontrolu interesantna je konstanta k_{Tr}=1/Tr, tj. cela struktura pogona indirektnog vektorskog upravljanja zavisi od tačnosti određivanja parametra k_{Tr}.

Izvršena je simulacija promene parametra k_{Tr} kod motora koji je opterećen opterećenjem m=1.5[Nm]. Motor se pokreće iz stanja mirovanja i u trenutku ***t=4s*** parametar k_{Tr} se poveća za 20%, dok se u trenutku ***t=8s*** smanji za 20% u odnosu na tačnu vrednost. Na osnovu rezultata sa Sl.4. može se zaključiti da se orijentacija sinhrono rotirajućeg koordinatnog sistema gubi sa promenom parametra k_{Tr} i da dolazi do gubitka vektorske kontrole što se ogleda u pojavi fluksa rotora po poprečnoj osi (ψ_{qr}).

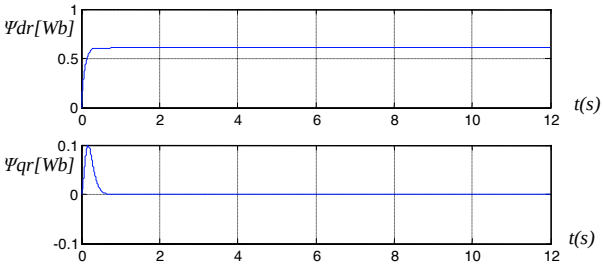
Pri gubljenju vektorske kontrole može se uočiti izrazita promena poprečne komponente struje statora, kao posledica promene parametra k_{Tr}. U slučaju da je motor neopterećen promena parametra k_{Tr} ne utiče na flukseve motora a nema uticaja ni na poprečnu komponentu struje statora.



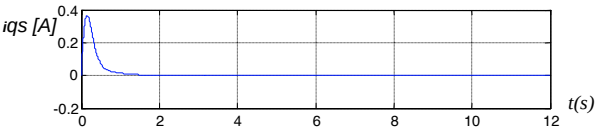
Sl.4. Simulacioni rezultati promene flukseva opterećenog motora sa 1.5[Nm] pri promeni kTr (±20%)



Sl.5. Simulacioni rezultat promene i_{qs} opterećenog motora sa 1.5[Nm] pri promeni parametra kTr (±20%)



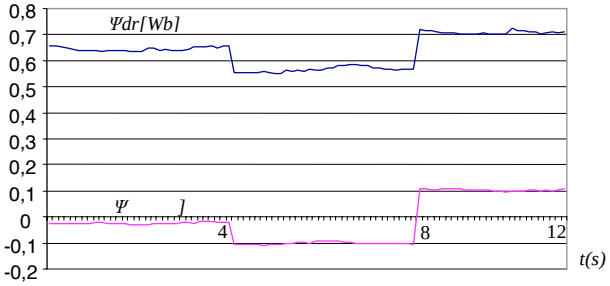
Sl.6. Simulacioni rezultati promene flukseva neopterećenog motora pri promeni parametrta kTr (±20%)



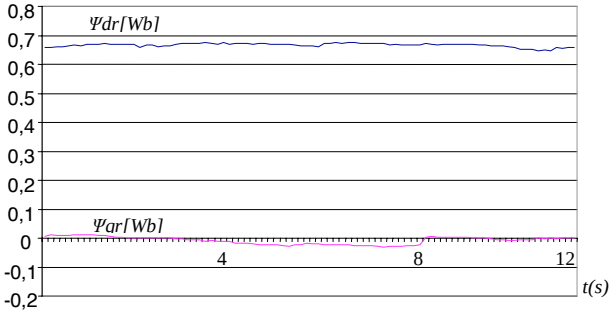
Sl.7. Simulacioni rezultat promene i_{qs} neopterećenog motora pri promeni parametrta kTr (±20%)

5. EKSPERIMENTALNA VERIFIKACIJA

Eksperimenti su vršeni na motoru čiji podaci su korišćeni u matematičkom modelu prilikom simulacije u programskom paketu *Matlab/Simulink.* Primenom Freescale DSP 56F8013 kao upravljačke jedinice praktično su potvrđeni simulacioni rezultati. Rezultati eksperimenta su prikazani na slikama Sl.8. i Sl.9.



Sl.8. Eksperimentalni rezultati promene flukseva opterećenog motora sa 1.5[Nm] pri promeni kTr (±20%)



Sl.9. Eksperimentalni rezultati promene flukseva neopterećenog motora pri promeni kTr (±20%)

6. ZAKLJUČAK

U radu je pokazana značajna osetljivost IFOC pogona na grešku parametra rotorskog kola. Takođe je pokazano da se ova osetljivost uvećava sa porastom opterećenja. Dakle, praćenje ovog parametra u toku rada pogona je neophodno i treba da se vrši kako sa promenom nivoa zasićenja magnetnog kola mašine tako i sa promenom temperature. S obzirom da se ostali uticaji ne mogu unapred predvideti, potrebno je ugraditi mehanizam za on-line (u toku rada pogona) identifikaciju vremenske konstante rotora.

7. LITERATURA

[1] V. Vučković, *Električni pogoni*, Akademska misao, 2002.
[2] G. Rašković, *Određivanje zagrejanja i prevalnog momenta trofaznog asinhronog kaveznog motora*, Novi Sad: Fakultet teničkih nauka ,diplomski rad, 1989.
[3] V. Vučković, *Opšta teorija el. mašina*, Nauka, 1992.

Kratka biografija:

Timer Janoš je Diplomski-master rad odbranio na FTN-u iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine u oktobru 2008.godine.

Evgenije Adžić rođen je u Subotici 1981. god. Magistrirao na Fakultetu Tehničkih nauka 2007. god., gde radi kao asistent.

Vlado Porobić rođen je u Bačkoj Palanci 1974. god. Magistrirao na Fakultetu Tehničkih nauka 2005. god., gde radi kao asistent.

Darko Marčetić rođen je u Novom Sadu 1968. god. Na ETF u Beogradu doktorirao 2006. god. Od 2007. radi kao docent na FTN, Novi Sad.

PRILAGOĐENJE NEWLIB BIBLIOTEKE GCC PREVODIOCU I POTREBAMA OPERATIVNIH SISTEMA U REALNOM VREMENU

Gvozden Nešković, *Fakultet tehničkih nauka, Novi Sad, gvozden.neskovic@micronasnit.com*
Milan Savić, *MicronasNIT, Institut za Informacione Tehnologije, Novi Sad, milan.savic@micronas.com*
Tatjana Aleksić, *MicronasNIT, Institut za Informacione Tehnologije, Novi Sad, tatjana.aleksic@micronas.com*

Sadržaj – *Rad daje pregled neophodnih izmena i prilagođenja newlib biblioteke za rad u namenskom sistemu. Posebna pažnja posvećena je prilagođenju biblioteke operativnom sistemu u realnom vremenu sa ciljem omogućavanja istovremenog poziva funkcija (reentrancy) biblioteke. Takođe, dat je primer realizacije jednostavnog ulazno-izlaznog sistema na kom se zasniva prilagodni deo newlib biblioteke.*

1. UVOD

Stalni trend razvoja informacionih tehnologija i računarske tehnike uzrokovao je pojavu inteligentnih kućanskih uređaja. Jednostanvi uređaji sa analognim elektronskim sklopovima postaju računarski sistemi sa centralnim procesorom, radnom memorijom i ulazno-izlaznim podsistemom. Ovaj trend je izražen je kod TV uređaja pojavom standarda za prenos signala u digitalnom obliku, kao i proširivanjem spektra usluga koje se nude korisniku. Povećanje kompleksnosti programske podrške i potreba za interaktivnim radom sa korisnikom uslovili su korišćenje operativnog sistema. Upotreba usluga operativnog sistema olakšava razvoj i skraćuje vreme potrebno za realizaciju i verifikaciju programske podrške. Specifičnosti TV platforme, kao što su dekodovanje slike i zvuka, postavljaju zahteve za strogo definisanje vremena odziva programske podrške na spoljne događaje. Ovi zahtevi uslovljavaju korišćenje operativnog sistema u realnom vremenu (Real Time Operating System - RTOS).

Kada se govori o razvoju programske podrške za namenske sisteme podrazumeva se upotreba programskog prevodioca koji se izvršava na PC računaru, a dobijeni mašinski kod je specifičan za ciljnu fizičku platformu (cross-compilation).

2. GCC

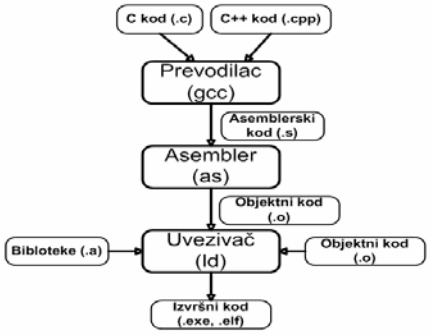
GCC kolekcija prevodilaca je skup prevodilaca za različite programske jezike. GCC je prihvaćen kao standardni prevodilac u modernim operativnim sistemima (Linux, BSD, Mac OS X). Iako je u početku bio namenjen za razvoj programske podrške za PC platformu, GCC je prilagođen za mnoge procesore koji se koriste u namenskim sistemima. GCC prevodilac je deo skupa alata za razvoj programske podrške (toolchain), koji se osim GCC prevodioca sastoji od asemblera i uvezivača koji se isporučju u okviru paketa pod nazivom binutils. Osnovni alati paketa binutils su asembler (as) i uvezivač (ld). Pored njih, tu se nalaze i pomoćni alati, kao što je ar – koristi se za rad sa arhivama objektnog koda i objcopy – koristi se za kopiranje i konverziju objektnih i izvršnih datoteka.

GCC prevodilac ima modularan oblik i sastoji se od:
- prednjeg dela,
- srednjeg dela i
- zadnjeg dela.

Prednji deo GCC prevodioca (front-end) sadrži module koji su specifični za određeni programski jezik. Ovi moduli prevode iskaze programskog jezika u interni prikaz koji je nezavisan od sintakse programskog jezika. Ovakva organizacija olakšava dodavanje podrške za novi programski jezik realizacijom odgovarajućeg modula u okviru prednjeg dela prevodioca. Neki od jezika koji su podržani u GCC prevodiocu su: C, C++, Objective-C, Java, ADA, FORTRAN i drugi.

Srednji deo (middle-end) prevodioca obavlja optimizacije koda koje su nezavisne od ciljne fizičke platforme. Neke od tehnika koje se primenjuju nad kodom su: uprošćavanje algebarskih izraza, izračunavanje konstantnih izraza i izbacivanje suvišnih (redudantnih) izračunavanja izraza. Zavisno od potreba, moguće je optimizovati objektni kod da se izvršava što brže, ili da zauzima što manje memorije za smeštanje.

Zadnji deo prevodioca (back-end) generiše asemblerski kod specifičan za ciljnu platformu. GCC prevodilac podržava veliki broj fizičkih platformi među kojima su: x86, x86-64, PowerPC, MIPS, ARM i druge.



Sl.1. *Faze prevođenja koda programske podrške*

Nakon što se izvorni kod programske podrške prevede u asemblerski oblik upotrebom GCC prevodioca, sledi faza generisanja objektnog koda za ciljnu platformu upotrebom asemblera (as). Nakon asembliranja dobijeni objektni kod uvezuje se sa bibliotekama i drugim objektnim datotekama upotrebom uvezivača (ld). Na slici 1. prikazane su faze prevođenja izvornog programskog koda do izvršnog mašinskog koda ciljne fizičke platforme upotrebom skupa alata za razvoj programske podrške.

Napomene:
a) Rad je proistekao iz diplomskog-master rada Gvozdena Neškovića. Mentor je bio prof.dr Nikola Teslić.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

3. C BIBLIOTEKA - NEWLIB

Newlib je C biblioteka namenjena za koriććenje u namenskim sistemima, razvijena kao zamena za standardnu glibc biblioteku. Najveća mana glibc biblioteke je njena veličina, koja nije pogodna za namenske sisteme koji imaju ograničenu veličinu radne memeorije i skromnu procesorsku moć. Pored toga, glibc sadrži veliki broj funkcija koje nisu potrebne za razvoj programske podrške za namenske sisteme. Newlib je kolekcija programskog koda iz različitih izvora (razvijena od strane Cygnus Solutions, sada Red Hat), koja se održava kao jedinstvena celina. Distribuira se pod licencom koja dozvoljava njenu upotrebu u komercijalnim namenskim sistemima.

Postoje i druge C biblioteke za korišćenje u namenskim sistemima. Najpoznatije su: uClibc, diet libc i sde libc. uClibc je C biblioteka je razvijena za potrebe CLinux operativnog sistema za namenske sisteme, što otežava njeno prilagođavanje drugim sistemima. Diet libc ima male memorijske zahteve, ali i ograničen broj realizovanih funkcija što ograničava njenu upotrebu u složnim sistemima. SDE libc je C biblioteka koja se isporučuje sa MIPS kolekcijom razvojnih alata, namenjenih isključivo za rad sa MIPS fizičkom platformom.

Newlib biblioteka sadrži realizaciju funckija za rad sa standardnim ulazom i izlazom (stdio). Newlib sadrži realizaciju dinamičkog alokatora memorije (funkcije malloc, free, realloc). Podržane su i standardne funkcije POSIX sprege za rad sa ulazno-izlaznim uređajima (open, write, read, close)[3]. Ove funkcije direktno se oslanjaju na funkcije prilagodnog dela (code stubs) ulazno-izlaznog podsistema korišćene platforme. Zaseban deo newlib biblioteke čini realizacija matematičke biblioteke saglasne sa IEEE standardom [4], pod nazivom libm. Libm, pored stadardnih matematičkih funkcija koje rade sa razlomljnim brojevima u dvostrukoj preciznosti, sadrži i funkcije koje rade sa razlomljenim brojevima u jednostrukoj preciznosti (float). Iako nisu podržane IEEE standardom, ove funkcije mogu biti korisne pri radu na fizičkim platformama skromnijih mogućnosti. Na slici 2. prikazan je sadržaj newlib biblioteke.

Sl.2. *Dijagram newlib biblioteke*

4. PRILAGOĐENJE NEWLIB BIBLIOTEKE

Prilikom prilagođenja newlib biblioteke potrebno je rešiti problem istovremenog poziva funkcija (reentrancy) newlib biblioteke i realizovati prilagodni deo za rad sa uslugama operativnog sistema i ulazno-izlaznog podsistema korišćene platforme [2]. Mogućnost istovremenog poziva funkcija je osobina biblioteke koja dozvoljava da više programskih niti poziva istu funkciju istovremeno. Korišćena programska podrška koristi ThreadX, komercijalni operativni sistem u realnom vremenu, namenjen za upotrebu u namenskim sistemima. Odeljak A opisuje izmene ThreadX RTOS-a neophodne za obezbeđivanje istovremenog poziva funkcija biblioteke iz više programskih niti. Odeljak B daje opis realizacije prilagodnog dela newlib biblioteke uz pomoć jednostavnog sistema za rukovanje ulazno-izlaznim uređajima.

A. OBEZBEĐIVANJE ISTOVREMENOG POZIVA FUNKCIJA NEWLIB BIBLIOTEKE

Većina funkcija newlib biblioteke projektovane su da omoguće istovremene pozive. Međutim, postoje funkcije koje ne mogu biti jednostavno realizovane za istovremeno korišćenje u višenitnom okruženju. ANSI C standardom propisana je upotreba globalne promenljive pod nazivom errno u koju se upisuje kod greške. Ovakva signalizacija greške nije moguća u sistemu sa više programskih niti koje pozivaju usluge biblioteke, jer je nemoguće utvrditi koja nit je proizvela grešku.

Problem globalnih promenljivih newlib rešava definisanjem globalnog pokazivača na strukturu tipa _reent koja sadrži sve globalne promenljive koje su propisane ANSI C standardom. Newlib biblioteka definiše globalni pokazivač na promenljivu tipa struct _reent pod nazivom _impure_ptr. Prilikom iniciranja biblioteke, kreira se jedna promenljiva tipa _reent strukture a pokazivač _impure_ptr se inicijalizuje adresom ove promenljive. Time je obezbeđeno da funkcije biblioteke ispravno koriste globalne promenljive u situaciji kada nema više programskih niti u sistemu.

Newlib biblioteka za sve funkcije prilagodnog dela pruža dve vrste programske sprege. Prva grupa funkcija ima standardno ime (npr. write) i podrazumevano koristi pokazivač _impure_ptr koji pokazuje na globalni objekat tipa _reent. Ove funkcije ne omogućavaju istovremeno pozivanje funkcija newlib biblioteke od strane više programskih niti jer koriste globalni objekat _reent strukture. Druga vrsta funkcija prilagodnog dela u listu parametara uključuju pokazivač na strukturu tipa _reent u kojoj se čuvaju vrednosti globalnih promenljivih za izvršavanu programsku nit. Upotrebom promenljive tipa _reent strukture omogućava se istovremeno pozivanje funkcija newlib biblioteke, jer svaki poziv funkcije biblioteke iz različitih programskih niti, koristi podatke koji su specifični za programsku nit. Ime ovih funkcija prilagodnog dela se formira dodavanjem prefiksa '_' i sufiksa '_r' na naziv funkcije (npr. _write_r).

Na osnovu navedenih karakteristika newlib biblioteke, istovremeno korišćenje bibliotečkih funkcija može se ostvariti na dva načina:

- Realizacijom funkcija prilagodnog dela newlib biblioteke koje u listu parametara uključuju pokazivač na lokalnu promenljivu tipa _reent. Za svaku nit u sistemu potrebno je kreirati jednu promenljivu tipa _reent.
- Realizacijom funkcija prilagodnog dela newlib biblioteke koje koriste pokazivač _impure_ptr za rad

sa globalnim promenljivma. Sve programske niti pri stvaranju iniciraju lokalnu promenljivu tipa _reent, čija se adresa postavlja u globalni pokazivač _impure_ptr svaki put kada operativni sistem izvršava programsku nit.

Da bi se ostvarila mogućnost istovremenog poziva funkcija newlib biblioteke realizacijom funkcija koje koriste _impure_ptr pokazivač za rad sa globalnim promenljivim, izmenjen je ThreadX RTOS. Izvršene su sledeće izmene u okviru ThreadX RTOS-a:

- U strukturu TX_THREAD, koja opisuje programsku nit ThreadX RTOS-a, uključeno je polje tipa _reent strukture.
- Prilikom stvaranja nove programske niti, u funkciji tx_thread_create, vrednost promenljive tipa _reent postavlja se na početnu vrednost.
- Pri prevođenju programske niti u stanje izvršavanja, u funkciji tx_thread_schedule, globalni pokazivač _impure_ptr newlib biblioteke postavlja se da pokazuje na promenljivu tipa _reent programske niti kojoj se dodeljuje procesor.

Navedene izmene operativnog sistema moguće je izvesti samo ako se raspolaže izvornim kodom korišćenog operativnog sistema. Međutim, to nije uvek slučaj kod upotrebe komercijalnih operativnih sistema.

B. REALIZACIJA PRILAGODNOG DELA NEWLIB BIBLIOTEKE

Funkcionalnost newlib biblioteke zasniva se na prilagodnom delu. Funkcije newlib biblioteke oslanjaju se na funkcije prilagodnog dela kad pristupaju sistemu datoteka, rade sa ulazno-izlaznim uređajima ili koriste usluge operativnog sistema kao što je kreiranje i uništavanje procesa. Ukoliko je operativni sistem saglasan sa POSIX 1003.1 standardom [3], većina funkcija prilagodnog dela je već realizovana unutar operativnog sistema. Da bi se newlib biblioteka mogla uvezati sa ostatkom objektnog koda sve funkcije prilagodnog dela moraju biti realizovane. U nastavku su navedene funkcije prilagodnog dela newlib biblioteke sa kratkim opisom:

- *void _exit(int)* Funkcija prekida izvršavanje programa.
- *void close(int fd)* Funkcija zatvara datoteku na koju se odnosi parametar fd.
- i*nt execve(char *name, char **argv, char **env)* Funkcija prebacuje kontrolu na novi proces.
- *int fork()* Funkcija kreira novi proces.
- *int fstat(int fd, struct stat *st)* Funkcija vraća status otvorene datoteke.
- *int getpid()* Funkcija vraća jedinstveni identifikator procesa.
- *int isatty(int fd)* Funkcija proverava da li je izlazna datoteka identifikovana parametrom fd terminal.
- *int kill(int pid, int sig)* Funkcija šalje signal navedenom procesu.
- *int link(char *old, char *new)* Funkcija menja ime postojećoj datoteci.
- *int lseek(int fd, int ptr, int dir)* Funkcija postavlja trenutnu poziciju unutar navedene datoteke.
- *int open(const char *name, int flags, int mode)* Funkcija otvara datoteku sa navedenim imenom, za namenu određenu parametrima flags i mode.
- *int read(int file, char *ptr, int len)* Funkcija čita iz navedene datoteke.
- *caddr_t sbrk(int incr)* Funkcija povećava veličinu dinamičke memorije za broj bajtova naveden parametrom incr.
- *int stat(char *file, struct stat *st)* Funkcija vraća status datoteke označene imenom.
- *int times(struct tms *buf)* Funkcija vraća statističke podatke prikupljene za trenutni proces.
- *int unlink(char *name)* Funkcija briše datoteku sa navedenim imenom.
- *int wait(int *status)* Funkcija čeka na završetak procesa.
- *int write(int fd, char *ptr, int len)* Funkcija upisuje podatke u datoteku.

Operativni sistemi u realnom vremenu, najčešće umesto koncepta procesa podržavaju rad sa nitima, što je slučaj i kod ThreadX RTOS-a. Ovo ograničenje je uvedeno zbog dodatne obrade pri smeni procesa čime se povećava vreme odziva sistema na spoljne događaje. Prema tome, funkcije newlib biblioteke za rad sa procesima ne mogu direktno da se preslikaju na usluge ThreadX RTOS-a. Realizacija ovih funkcija vraća kod greške, dok je programska sprega za rad sa nitima definisana u ThreadX RTOS-u.

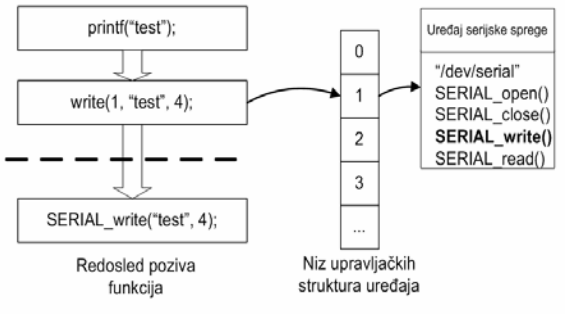
Newlib funkcije koje rade sa standardnim ulazom i izlazom oslanjaju se na funkcije prilagodnog dela za rad sa upravljačima uređaja (device drivers) i usluge sistema datoteka (file system). Iako većina namenskih sistema nema realizovan sistem datoteka, od velike važnosti je mogućnost generisanja izlaza, npr. upotreba serijske sprege platforme za potrebe praćenja toka izvršavanja i otkrivanje grešaka. Funkcije newlib biblioteke pozivaju funkciju write svaki put kad šalju podatke uređaju ili vrše ispis na standardni izlaz. Prvi parametar funkcije write označava uređaj kome se podaci šalju i može da bude bilo koji celobrojan broj. Jedino ograničenje koje unosi newlib biblioteka je da se uređaj sa oznakom 0 tretira kao standardni ulaz (stdin), uređaj sa oznakom 1 kao standardni izlaz (stdout) a uređaj sa oznakom 2 kao standardni izlaz za greške (stderr). Da bi realizacija ovih funkcija bila moguća, mora postojati koncept upravljanja uređajima koji definiše skup osnovnih operacija za rad sa uređajom. Svakom uređaju dodeljuje se jedinstveno simboličko ime koje je moguće preslikati u jedinstveni identifikator.

Realizacijom upravljača uređaja [2], upotrebom strukture koja opisuje svojstva i funkcije za rad sa uređajom, pojednostavljen je rad sa ulazno-izlaznim uređajima. Struktura koja opisuje svojstva uređaja sadrži sledeća polja:

- Naziv uređaja (name),
- Pokazivač na funkciju za otvaranje uređaja (open),
- Pokazivač na funkciju za zatvaranje uređaja (close),
- Pokazivač na funkciju za upis podataka (write),
- Pokazivač na funkciju za čitanje podataka (read).

Pri iniciranju uređaja, kreira se nova promenljiva navedene strukture koja se smešta u niz kreiranih uređaja. Identifikator uređaja koristi se kao indeks niza struktura koje opisuju uređaje. Pozivom funkcije za čitanje ili upis uređaja obavlja se prenos podataka između funkcija newlib biblioteke i ulazno-izlaznih uređaja. Na slici 3. prikazan je mehanizam kojim se podaci za ispis na standardni izlaz, korišćenjem

funkcije printf, prosleđuju funkciji za upis podataka serijske sprege.



Sl.3. *Ispis na standardni izlaz*

Realizacija funkcije read analogna je realizaciji funkcije write. Funkcija open, koja kao parametar prima simbolički naziv uređaja, u nizu kreiranih uređaja pronalazi uređaj čiji naziv odgovara navedenom parametru i poziva funkciju open upravljača odgovajućeg uređaja. Indeks niza, koji odgovara strukturi pronađenog uređaja, vraća se kao identifikator uređaja.

Funkcije upravljača uređaja koriste sinhronizacine promenljive ThreadX RTOS-a da obezbede atomičnost operacija nad uređajima. Na ovaj način se tokovi podataka koji potiču iz različitih programskih niti ne prekidaju dok se u celosti ne završi započeta operacija. Ovaj model rešavanja isključivosti rada zajedničkih resursa (ulazno-izlazni uređaji) korišćen je pri realizaciji bibliotečkih funkcija kako bi newlib biblioteka ostala nezavisna od okruženja i operativnog sistema sa kojim se koristi.

Svaki put kada se povećava količina memorije koja se dinamički dodeljuje programskoj niti, newlib poziva funkciju sbrk. Parametar ove funkcije određuje za koliko se povećava memorija koja se koristi za dinamičku dodelu. Memorija rezervisana za dinamičku dodelu nalazi se na kraju memorijske mape, definisane uvezivačkom skriptom koja se koristi u procesu uvezivanja. Uvezivačka skripta prosleđuje adrese početka i kraja ove memorijske zone, na osnovu kojih se realizuje navedena funkcija. Kako dinamička dodela memorije zavisi od globalnog resursa koji koriste sve programske niti (radne memorije), neophodno je realizovati zaštitu u vidu isključive sekcije. Funkcije __malloc_lock i __malloc_unlock, koje newlib koristi u tu svrhu, realizovane su upotrebom mjuteks sinhronizacionih promenljivih ThreadX RTOS-a.

5. TESTIRANJE

Korektnost opisane realizacije verifikovana je skupom ispitnih slučajeva. Ispitni slučajevi koriste funkcije biblioteke koje se, bez navedenih modifikacija, ne mogu pozivati iz više programskih niti. Ispitini slučajevi su realizovani korišćenjem programskih niti koje pozivaju iste bibliotečke funkcije. Pored ispitivanja korektnosti funkcija, realizovan je poseban ispitni slučaj koji proverava unikatnost errno promenljive za svaku programsku nit.

Izmenom funkcije tx_thread_schedule ThreadX-a povećano je kašnjenje pri smeni niti za 10 ns. Kašnjenje je mereno korišćenjem brojačkog registra koprocesora CP0 MIPS centralnog procesora familije 24K [5], koji radi na 300 MHz. Pre izmene ovo kašnjenje je iznosilo 183 ns, a nakon izmene 193 ns. Uneseno povećanje kašnjenja pri smeni niti ThreadX RTOS-a nije kritično za ciljni namenski sistem.

6. ZAKLJUČAK

Rad daje pregled korišćenja programskih alata za prevođenje i uvezivanje programske podrške za namenske sisteme upotrebom GCC skupa prevodilaca i newlib biblioteke kao C standardne biblioteke. Opisani skup alata predstavlja dobru polaznu osnovu za realizaciju programske podrške za namenske sisteme.

Zamenom algoritma za dinamičku dodelu memorije moguće je dodatno prilagoditi newlib biblioteku potrebama ciljne platforme. Upotreba dinamičkog alokatora memorije na fizičkim platformama koje imaju malu količinu radne memorije često je neopravdana zbog fragmentacije koju ovaj algoritam unosi [6]. U ovom slučaju bolje iskorištenje memorije postiže se upotrebom algoritma blokovske dodele memorije, koji je često realizovan u sklopu RTOS-a.

Prikazano rešenje moguće je lako prilagoditi različitim platformama što je jedan od bitnih uslova pri odabiru skupa alata za razvoj programske podrške za namenske sisteme.

7. LITERATURA

[1] Literatura sa predavanja iz predmeta "Programska podrška u televiziji i obradi slike," FTN Novi Sad 2005

[2] B. Gatliff, "Embedding GNU: Newlib," *Embedded System Design, www.embedded.com,* Mart 2002

[3] IEEE Stadard 1003.1, 2004 Edition, *www.opengroup.com*

[4] IEEE Standard for Binary Floating-Point Arithmetic ANSI/IEEE Std 754, 1985

[5] Dominic Sweetman, "See MIPS Run, Second Edition," Morgan Kaufmann Publishers, Oktobar 2006

[6] M. Barr, A. Massa, "Programming Embedded Systems," O'Reilly, Oktobar 2006

Abstract – *Thi*s paper presents use of newlib C runtime library in embedded system. Special attention is paid to porting newlib library to platform's I/O system calls and achieving reentrancy of library functions in multithreading environment of real time operating system.

PORTING NEWLIB LIBRARY FOR USE WITH GCC COMPILER AND REAL TIME OPERATING SYSTEM

Gvozden Nešković, Milan Savić, Tatjana Aleksić

JEDAN KONCEPT OPTIMIZACIONE TEHNIKE ZA ISKORIŠĆENJE ADRESNE JEDINICE U C KOMPAJLERU

Zoran Zarić, Miodrag Đukić, Marko Gajić, Miroslav Popović, *Univerzitet u Univerzitet u Novom Sadu,Fakultet tehničkih nauka, Odsek za računarsku tehniku i računarske komunikacije*

Sadržaj – *U ovom radu je opisana optimizaciona tehnika koja povećava iskorišćenost jedinice za generisanje adresa procesora Coyote 32. Povećanje iskorišćenosti jedinice za generisanje adresa biće postignuto tako što će se tokom kompajliranja promeniti indeksno adresiranje elemenata niza u posredno adresiranje. Ova optimizaciona tehnika biće primenjena samo u slučajevima u kojim se indeks ažurira u koracima ±1 i ±2. Analize pokazuju da bi se implementacijom ove optimizacione tehnike dobio značajno poboljšan asemblerski kod.*

1 UVOD

Digitalni signal procesori se koriste kao ciljna platforma pri realizaciji algoritama za obradu digitalnih signala u realnom vremenu. U tu grupu spadaju algoritmi kao što su kodeci, filtri, FFT–ovi i slično. Pisanje ovakvih algoritama za ciljnu platformu predstavlja težak i skup posao i zbog toga je mnogo isplativija i brža realizacija ovih algoritama u nekom višem programskom jeziku. Naravno, da bi ovako nešto bilo moguće, prvo mora postojati kompajler koji će prevesti kod pisan u višem programskom jeziku i prilagoditi ga ciljnoj platformi.

Jedan takav C kompajler je razvijen na Odseku za računarsku tehniku i računarske komunikacije Fakulteta tehničkih nauka u Novom Sadu i zove se Cirrus C Compailer (ili CCC). Ciljna platforma ovog kompajlera je digitalni signal procesor Coyote 32 kompanije Cirrus Logic. U ovom radu je predložena optimizaciona tehnika koja će kompajleru omogućiti bolje iskorišćenje adresnog generatora prilikom prevođenja funkcija koje sadrže pristup elementima niza preko indeksa. Optimizacija se odnosi samo na ažuriranje indeksa u koracima od ±1 i ±2. Korisnost ove optimizacije neposredno zavisi od učestalosti pojave naredbi koje pristupaju elementima niza preko indeksa u aplikaciji koja se prevodi. Analizirane su dve aplikacije i analize pokazuju da je dobit značajna.

2. ARHITEKTURA COYOTE 32 PROCESORA

Coyote 32 procesor je digitalni signal procesor (DSP) sa nepokretnim zarezom i izraženom protočnom arhitekturom sa dva jezgra. Ovaj DSP procesor spada u grupu procesora sa dugačkom instrukcijskom reči.

Svako jezgro Coyote 32 procesora poseduje svoju lokalnu memoriju u kojoj se nalazi programski memorijski prostor i memorijski prostor za podatke. Memorijski prostor za podatke je podeljen na X i Y deo. Svakom od ovih delova se može pristupiti neposrednim adresiranjem 6–bitnom adresom ili korišćenjem adresnih registara. Omogućen je pristup i X i Y zoni odjednom, ali uz ograničenja. Detaljnije objašnjenje pomenutih ogrničenja može se naći. u [1].

Jezgro Coyote 32 sastoji se iz tri dela:

- jedinica za kontrolu toka programa,
- dve paralelne jedinice za generisanje adresa i
- dva paralelna toka podataka

Svaka jedinica za generisanje adresa sadrži četiri 16–bitnih adresnih registara. Za jednu jedinicu za generisanje adresa vezani su adresni registri od i0 do i3, a za drugu adresni registri od i4 do i7. Postoje još četiri adresna registra (od i8 do i11), ali oni su rezervisani za rad operativnog sistema Coyote 32 platforme i nisu dostupni programeru. Pored adresnih registara, postoji i osam 16–bitnih nm registara koji omogućavaju promenu režima posrednog adresiranja odgovarajućeg adresnog registra.

Postoje tri režima posrednog adresiranja:

- linearno adresiranje,
- moduo adresiranje i
- bit-inverzno adresiranje

Arhitektura Coyote 32 jezgra omogućava ažuriranje adresnih registara u istoj instrukciji u kojoj su i korišćeni za pristup memoriji. Ažuriranje je dozvoljeno u koracima ±1, ±2 i ±n, gde n predstavlja 12–bitni broj koji se pre toga mora smestiti u prvih dvanaest bita nm registra, koji odgovara korišćenom adresnom registru.

3. OPIS PROBLEMA

Današnji digitalni signal procesori poseduju složene jedinice za generisanje adresa, ali većina kompajlera ih jako slabo iskoristi. Razlog ovoga je u činjenici što kompajler najčešće ne analizira funkcionalan smisao skupova naredbi. Na slici 1 je prikazana funkcija *primer1* čiji je zadatak da sabere sve elemente dva niza:

```
int primer1()
{
    int i, suma = 0;
    for (i = 0 ; i < 10 ; i++)
            suma += niz1[i] + niz2[i];
    return suma;
}
```

Slika 1. *Primer funkcije u kojoj se nizovima pristupa indeksno*

Funkcija *primer1* sabira sve elemente oba niza u promenljivu suma, pri čemu za indeksiranje oba niza koristi indeksnu promenljivu i. Ovaj način pristupa elementima niza je standardan u C jeziku, ali takav način adresiranja nije podržan od strane Coyote 32 procesora. Kada se funkcija *primer1* prevede upotrebom CCC–a dobije se asemblerski kod prikazan na slici 2.

Napomene:

a) Rad je proistekao iz diplomskog-master rada Zorana Zarića. Mentor je bio prof.dr Miroslav Popović.

b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

```
_primer1
            uhalfword(a0) = (0)          # promenljiva suma
            uhalfword(b1) = (0)          # promenljiva i
            do (10), Boundary            # početak for petlje
            b0 =+ b1
            i0 = (0) + (_niz1)           # adresa niza niz1
            AnyReg(a1, i0)
            b0 = b0 + a1                 # evaluacija adrese elementa
            i0 = b0;
            nop
            a2 = xmem[i0]                # čitanje iz niza niz1
            a1 =+ b1
            i0 = (0) + (_niz2)           # adresa niza niz2
            AnyReg(b0, i0)
            a1 = a1 + b0                 # evaluacija adrese elementa
            i0 = a1
            nop
            a1 = xmem[i0]                # čitanje iz niza niz2
            a1 = a1 + a2                 # sabiranje elemenata nizova
            a1 = a1 + a0                 # sabiranje sa sumom
            a0 =+ a1
            uhalfword(a1) = (1)
            a1 = a1 + b1                 #inkrementiranje indeksa
Boundary:   b1 =+ a1
__primer1_END
            ret
```

Slika 2. *Dobijeni asemblerski kod prevodjenjem funkcije primer1*

Uočava se da dobijeni kod ne koristi prednosti Coyote 32 arhitekture. Razlog je u tome što je pristup elementima niza preko indeksa mnogo sličniji načinu adresiranja kod procesora opšte namene nego kod digitalnih signal procesora. Procesori opšte namene poseduju više načina adresiranja kao što su: neposredno adresiranje, posredno adresiranje, indeksno adresiranje, itd. Ovakvi načini adresiranja omogućavaju brzu realizaciju raznih struktura podataka i veoma su slični sa načinima adresiranja koji se pojavljuju u višim programskim jezicima. Digitalni signal procesori uglavnom poseduju manji broj načina adresiranja, jer je njihova namena specijalizovanija. Pošto Coyote 32 procesor poseduje samo dva načina adresiranja podataka (neposredni i posredni), prevodjenjem primera1 pomoću CCC–a dobijen je neefikasan asemblerski kod. Na sledećoj slici je prikazana funkcija kod koje se kreće kroz niz pomoću pokazivača na taj niz:

```
int primer2 ()
{
   int i, suma = 0;
   int* pniz1 = niz1;
   int* pniz2 = niz2;
   for (i = 0 ; i < 10 ; i++)
      suma += *pniz1++ + *pniz2++;
   return suma;
}
```

Slika 3. *Primer funkcije u kojoj se nizovima pristupa preko pokazivača*

Kao što se može videti, funkcionalne razlike između navedenih primera nema, pri čemu je kod poslednjeg primera neophodno uvesti još dva pokazivača. Kada se uporede funkcije *primer1* i *primer2* primećuje se da je *primer2* veći za tačno dve naredbe C jezika. Ove naredbe su neophodne radi kreiranja i pozicioniranja novih pokazivača na početak nizova *niz1* i *niz2*. Rezultat prevođenja funkcije *primer2* prikazan je na slici 4:

```
_primer2
            uhalfword(a0) = (0)          # promenljiva suma
            i1 = (0) + (_array1)         # adresa niza niz1
            i0 = (0) + (_array2)         # adresa niza niz2
            do (10), Boundary
            a1 = xmem[i1]; i1 += 1       # čitanje iz niza niz1
            b0 = xmem[i0]; i0 += 1       # čitanje iz niza niz2
            b0 = b0 + a1                 # sabiranje elem. nizova
            b0 = b0 + a0                 # sabiranje sa sumom
Boundary:   a0 =+ b0
__primer2_END
            ret
```

Slika 4. *Dobijeni asemblerski kod prevodjenjem funkcije primer2*

Slika pokazuje da pristupanjem nizovima pomoću pokazivača u C kodu aplikacije za rezultat ima znatno manji asemblerski kod, pri čemu se indeksiranje nizova obavlja samo uz pomoć adresnih registara, a određivanje sadržaja adresnih registara se vrši u istom ciklusu kao i indeksiranje. Razlog ovako manjeg broja asemblerskih instrukcija je sličnost načina adresiranja koji je korišćen u funkciji *primer2* sa načinom adresiranja koji omogućava jedinica za generisanje adresa Coyote 32 procesora. Ova sličnost olakšava posao kompajleru, jer je u mogućnosti da iskoristi postojeći posredan način adresiranja.

Na slici 5 su upoređeni delovi koda prevedenih funkcija koji predstavljaju jedno adresiranje elementa niza:

```
b0 =+ b1                    a1 = xmem[i1]; i1 += 1
 i0 = (0) + (_niz1)
AnyReg(a1, i0)
b0 = b0 + a1
i0 = b0;
nop
a2 = xmem[i0]
```

Slika 5. *Adresiranje elementa niza u prevedenim funkcijama: primer1(levo) i primer2 (desno)*

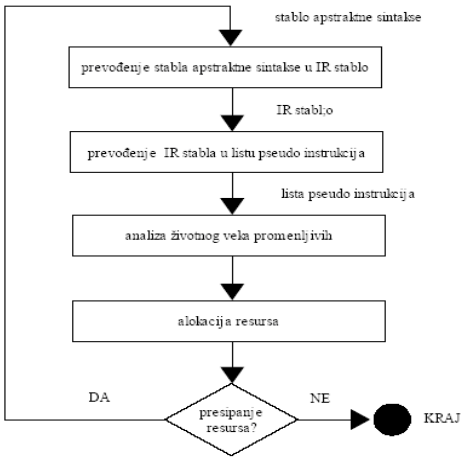
Prilikom indeksnog adresiranja elemenata niza u C kodu, dobije se tačno šest instrukcija više u asemblerskom kodu.

Prema tome, programer može uticati na kvalitet dobijenog asemblerskog koda tako što će pri pisanju aplikacije u višem programskom jeziku koristiti samo one načine adresiranja podataka koji su podržani od strane ciljne DSP platforme. Ovo je protivno ideji koja stoji iza viših programskih jezika, jer programer koji piše aplikaciju u nekom od viših programskih jezika ne mora biti detaljno upoznat sa arhitekturom ciljne platforme. U cilju poboljšanja izlaznog koda potreban je mehanizam unutar kompajlera koji će promeniti način adresiranja podataka u aplikaciji pisanoj u višem programskom jeziku tako da odgovara načinu adresiranja koji podržava ciljna platforma.

4. OPIS CIRRUS C KOMPAJLERA I MOGUĆNOST REALIZACIJE PREDLOŽENE OPTIMIZACIJE

Cirrus C kompajler se sastoji iz dva osnovna dela: prednjeg i zadnjeg. Prednji deo kompajlera obavlja leksičku i sintaksnu analizu C aplikacije. Za temu ovog rada dovoljno je pomenuti da se kao izlaz prednjeg dela kompajlera dobija aplikacija predstavljena stablom apstraktne sintakse. Dobijeno stablo apstraktne sintakse se dalje prosleđuje zadnjem delu kompajlera. Zadnji deo CCC–a se sastoji iz više modula, od kojih su za ovaj rad bitna samo četiri: modul koji prevodi stablo apstraktne sintakse u stablo medju reprezentacije (IR stablo), modul koji prevodi stablo

medjureprezentacije u listu pseudo asemblerskih instrukcija, modul koji analizira životni vek promenljivih i modul koji dodeljuje resurse. Na slici 6 je prikazan pojednostavljen blok dijagram CCC–a:



Slika 6. *Blok dijagram Cirrus C kompajlera*

Slika pokazuje da su moduli mađusobno linearno povezani i da obavljaju prevođenje u jednoj ili više iteracija, u zavisnosti da li je u poslednjoj iteraciji postojalo prelivanje resursa (*spill*) ili ne. Promenljive koje se ne mogu smestiti u traženu klasu resursa, prelivaju se u nižu klasu resursa i tada počinje nova iteracija prevođenja. Svako prelivanje resursa u stvari predstavlja fizičko premeštanje premenljive za koju nema mesta u traženoj klasi resursa. Istu je promenljivu potrebno vratiti u traženu klasu resursa onog trenutka kada se ona pojavi kao deo naredbe C jezika. Prema tome, svako prelivanje resursa u asemblerski kod unosi dodatan broj instrukcija.

Prevođenje stabla apstraktne sintakse u IR stablo je pravolinijsko zato što postoji jednoznačno preslikavanje između stabla apstraktne sintakse i IR stabla. Dobijeno IR stablo nije zavisno ni od arhitekture ciljne platforme, niti od višeg programskog jezika od koga je nastalo. Radi daljeg uprošćavanja IR stabla, nad njim se obavlja kanonizacija [2].

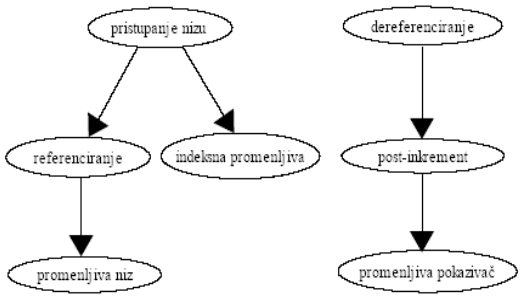
Da bi se IR stablo moglo prevesti u listu pseudo instrukcija potrebno je proceniti koji je najmanji broj pseudo instrukcija potrebnih da se pokrije celo IR stablo trenutne aplikacije. Ovo se procenjuje tako što se pseudo instrukcije organizuju u funkcionalne šablone, a zatim se određuje koji je to najmanji skup šablona koji bi mogao popločati celo IR stablo.

Analiza životnog veka promenljivih se svodi na analizu koja utvrđuje koje promenljive su žive u istom trenutku. Informacije dobijene analizom životnog veka promenljivih od krucijalne su važnosti pri dodeli resursa, zato što se samo na osnovu njih može odrediti koje promenljive mogu deliti isti fizički resurs. Detaljniji funkcionalan opis kompajlera može se naći u [2] i [3].

Pseudo asemblerske instrukcije se razlikuju od realnih asemblerskih instrukcija samo po tome što koriste apstraktne registre i akumulatore. Za razliku od realnih registara i akumulatora, apstraktnih registara i akumulatora ima neograničeno mnogo. To znači da će se krajnja lista realnih asemblerskih instrukcija razlikovati od liste pseudo asemblerskih instrukcija samo po imenima registara i akumulatora. Iz tog razloga je u CCC–u najveći broj optimizacionih tehnika primenjeno odmah posle prevodjenja IR stabla u listu pseudo asemblerskih instrukcija. Optimizaciona tehnika koja je analizirana u ovom radu, ne može se primeniti posle popločavanja IR stabla, zato što su tada asemblerske instrukcije već izabrane. U tom trenutku je jako teško odrediti koje instrukcije predstavljaju traženo adresiranje u aplikaciji koja se prevodi. Pošto IR stablo nije zavisno od višeg programskog jezika, najbolje mesto za realizaciju ove optimizacije je na stablu apstraktne sintakse.

Stablo apstraktne sintakse se sastoji samo od dva različita tipa elemenata: iskaz i izraz. Izraz predstavlja promenljivu i logičku ili aritmetičku operaciju nad jednim ili više izraza. Izraz se može sastojati od kompozicije drugih izraza. U iskaz spadaju programske petlje, naredbe grananja, blokovi naredbi, naredba povratka iz funkcije itd. Iskaz može biti sastavljen od kompozicije iskaza i izraza ili samo iskaza. Izuzetak je iskaz koji predstavlja blok naredbi, jer on u sebi sadrži samo listu drugih iskaza. Svaka funkcija koja čini sastavni deo aplikacije, koja se prevodi, predstavljena je iskazom bloka naredbi. Taj iskaz u sebi sadrži listu svih iskaza koji čine tu funkciju.

Na slici 7 su prikazana stabla apstraktne sintakse: indeksnog načina adresiranja elemenata niza i adresiranja elemenata niza pomoću pokazivača sa inkrementom od +1 (slično je sa inkrementom -1).



Slika 7. *Adresiranje niza pomoću indeksa (levo) i adresiranje niza pomoću pokazivača (desno)*

Na osnovu slike 7 se može zaključiti da je za promenu načina adresiranja elemenata dovoljno uvesti novi pokazivač istog tipa kao i niz. Da bi uvedeni pokazivač pokazivao na početak niza potrebno je dodati izraz stablu apstraktne sintakse koji to opisuje. Nakon toga, zameniti stablo koje opisuje indeksno adresiranje niza (leva strana slike), stablom koje opisuje indeksiranje niza uvedenim pokazivačem (desna strana slike). U slučaju da se isti element niza indeksira više puta u okviru iste iteracije, samo stablo apstraktne sintakse zadnjeg indeksiranja sme u sebi imati izraz za post-inkrementiranje niza. Sličan princip se može primeniti i na inkrement od ±2. Primer u kome bi se optimizovao slučaj za inkrement od ±n nije razmatran, zato što je utvtđeno da bi optimizacija tog tipa dala veoma slabe rezultate, a da je njena

realizacija vremenski veoma zahtevan posao. Opisana optimizaciona tehnika je pomenuta i u [4].

U radu [5] je opisana optimizaciona tehnika koja analizira kojim se redosledom pristupa elementima niza i na osnovu dobijenih informacija pokušava da smanji broj inkrementiranja indeksa za stopu veću od ±1. To se postiže uvođenjem dovoljnog broja pokazivača na elemente niza, tako da je za pristup sledećem traženom elementu niza uvek dovoljno inkrementiranje nekog od uvedenih pokazivača za stopu od ±1. Realizacija ove optimizacione tehnike je moguća u CCC-u, ali tek po realizaciji optimizacione tehnike opisane u ovom radu

5. ANALIZA IZABRANIH APLIKACIJA

Radi procene kolika bi bila stvarana dobit primene navedene optimizacije, analizirane su dve aplikacije: True Sorround HD i Volume IQ Control. Pomenute aplikacije su referentne aplikacije pisane u C kodu od strane programera kompanije SRS. Prema tome ove aplikacije nisu optinizovane za neku platformu.

Kao što je prikazano ranije, svako pristupanja elementima niza preko indeksa unosi dodatnih šest instrukcija. Prema tome, da bi se analizirala dobit primene opisane optimizacione tehnike, dovoljno je bilo prebrojati pojavu pristupa elementima niza preko indeksa u izabranim aplikacijama. Dobijeni broj je pomožen sa brojem dodatnih instrukcija i tako je izračunato očekivano smanjenje asemblerskog koda. U tabeli na slici 8 su prikazani rezultati dobijeni ovim prebrojavanjem, kao i očekivano smanjenje broja instrukcija u aplikacijama:

aplikacija	*broj pojava indeksnog adresiranja*	*očekivano smanjenje instrukcija*	*broj instrukcija pre optimizacije*	*smanjenje količine koda u procentima*
Volume IQ Control	44	264	2635	10.02
True Sorround HD	108	648	5534	11.71

Slika 8. *Očekivano smanjenje broja instrukcija u aplikacijama*

Treća kolona tabele pokazuje dobijeni broj instrukcija prevođenjem aplikacije pomoću CCC–a. Očekuje se i značajan pad u broju MIPS–a, iz razloga što je utvrđeno da je preko 90 odsto pojava indeksnog načina adresiranja vezano za sve iteracije obrade. Potrebno je spomenuti i očekivanu značajno manju pojavu prelivanja resursa, što za rezultat ima dalje smanjenje broja instrukcija. Na kraju, zaključeno je da je očekivana dobit, primenom opisane optinizacione tehnike, dovoljno velika da bi se isplatila njena realizacija na CCC–u.

LITERATURA

[1] *Coyote 32-bit DSP: Instruction Set and Architecture Reference Manual*, Cirrus Logic, 2005.

[2] V. Kovačević, M. Popović, Sistemska *programska podrška u realnom vremenu*, Fakultet tehničkih nauka, Novi Sad, 2002.

[3] A. V. Aho, R. Sethi, J. D. Ulman, *Compilers – principles, techniques, and tools*, Addison-Wesley, Massachusetts, 1986.

[4] M. R. Smith, *Code optimization techniques for dsp applications*, 9th IEEE DSP Workshop, Hunt, Texas, 2000.

[5] R. Leupers, A. Basu, P. Marwedel, *Optimized Array Index Computation in DSP Programs*, Asia South Pacific Design Automation Conference (ASP-DAC), 1998.

Abstract – This paper deals with compiler optimization technique that improves utilization of address generation unit at Coyote 32 processor. This will be done by changing the index addressing mode (for accessing array elements) to indirect addressing mode during compiling. The optimization technique is applied only to the cases with index modification by ±1 and ±2. The analysis shows that the implementation of this optimization technique results in greatly improved assembly code.

A CONCEPT OF OPTIMIZATION TECHNIQUE FOR BETER UTILIZATION OF ADDRESS GENERATION UNIT IN C COMPILER

Zoran Zarić, Miodrag Đukić, Marko Gajić, Miroslav Popović

PROJEKTOVANJE, IMPLEMENTACIJA I UPOREDNA ANALIZA RAZLIČITIH OBJEKTNIH MODELA CORBA KLIJENTSKIH APLIKACIJA

Ivana Popović, Fakultet Tehničkih Nauka, Novi Sad,
Vladimir Savić, DMS Group, Novi Sad,
Petko Milidragović, DMS Group, Novi Sad

Sadržaj – *U radu je opisana realizacija i uporedna analiza tri različita modela univerzalnih klijentskih aplikacija za testiranje CORBA servera koji su deo softverskog proizvoda kompanije DMS Group. Cilj ovog rada je primena različitih objekno orjentisanih metodologija u procesu projektovanja i implementacije svakog od prikazanih modela, a zatim i izbor modela koji najbolje ispunjava zahteve proširivosti i jednostavnosti upotrebe. Ukratko je opisana realizacija svakog od modela i dat je prikaz softverskih metrika korišćenih za potrebe uporedne analize.*

1. UVOD

Jezgro DMS softverskog paketa kompanije DMS Group čini DMS servis. On se sastoji iz više CORBA servera, čija je uloga preuzimanje i slanje podataka iz/ka SCADA-i i bazi podataka, kao i opsluživanje klijenata. U ovom radu predstavljena su 3 različita modela univerzalnih CORBA klijentskih konzolnih aplikacija, čiji je primarni zadatak testiranje, odnosno verifikacija rada serverskih komponenti.

U procesu posmatranja hijerarhije CORBA interfejsa (*.idl* datoteka) DMS servera, kao i analizom njihovih dijagrama klasa i aktivnosti, uočeno je da se funkcionalnost servera klijentima nudi preko odgovarajućih tipova sesija. Funkcionalnost koja je zajednička svim serverskim komponentama realizuje se kroz dva tipa sesija. To su osnovna (u daljem tekstu *BaseSession*), koju može dobiti svaki autorizovani korisnik, i menadžerska sesija (u daljem tekstu *ManagementSession*), koju dobijaju samo klijenti sa višim stepenom autorizacije. Minimalni zahtev koji se postavlja pred univerzalni model je da obezbedi upravo funkcionalnost za testiranje ova dva tipa sesija. Funkcionalnost specifična za pojedine serverske komponente obezbeđuje se takođe kroz sesije koje predstavljaju nadogradnju na osnovnu funkcionalnost (i realizuju se nasleđivanjem *BaseSession i ManagementSession*). Primarna svrha realizovanih univerzalnih klijentskih modela je standardizacija procesa konstrukcije klijentskih aplikacija, primenom koncepata objektno orjentisanog programiranja, pri čemu su osnovni zahtevi koji se postavljaju pred model univerzalne klijentske komponente pouzdanost, minimiziranje složenosti (radi jednostavnije upotrebe i korišćenja) i proširivost.

2. PROGRAMSKO OKRUŽENJE I SOFTVERSKI ALATI

Implementacija modela prikazanih u ovom radu realizovana je u *Visual Studio 2005* programskom okruženju, upotrebom programskog jezika *C++*. Za potrebe korišćenja *CORBA*-e upotrebljena je *omniORB 4.11*. biblioteka. Prikazane softverske metrike dobijene su korišćenjem *Resource Standard Metrics* (*Evaluation Version*) softverskog alata (http://msquaredtechnologies.com/m2rsm/).

3. MODEL M1

Prvi model nastao je kao rezultat preslikavanja CORBA interfejsa serverskih komponenti i njihove hijerarhije na odgovarajuće klase u okviru univerzalnog test klijent modela. Kao posledica toga, svaka klasa za testiranje određenog tipa sesije (u daljem tekstu test klasa) u okviru modela M1 formira se nasleđivanjem svih klasa za testiranje sesija od kojih je posmatrana sesija izvedena (što je omogućeno principima nasleđivanja, apstrakcije i polimorfizma). Znatno pojednostavljeni prikaz klasa test klijent aplikacije u koju je ugrađen univerzalni model M1 dat je na slici 1 (napomena: svaka test klasa u nazivu ima reč *helper*).

Radi jasnijeg prikaza, pojedinačno su predstavljene samo test klase za različite tipove sesija. Tamnijom nijansom uokvirene su klase koje su deo univerzalnog modela, a svetlijom klase koje predstavljaju nadogradnju funkcionalnosti za određeni server (oznaka SS u nazivu klase ima značenje *server specific*). Na slici su prikazane dve test klase za specifične sesije određenog servera, ali u opštem slučaju može ih biti proizvoljan broj. Neke od klasa koje obezbeđuju koordinaciju funkcionisanja test klasa i interaktivni rad deo su univerzalnog modela, dok su ostale deo nadogradnje specifične za konkretnu klijentsku komponentu (ove klase formiraju se nasleđivanjem klasa univerzalnog modela) .

Slika 1. *Šematski prikaz klijentske aplikacije sa ugrađenim univerzalnim modelom M1*

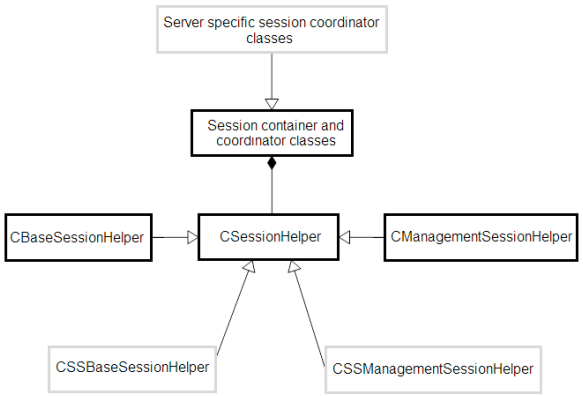
Odlika ovog modela je činjenica da je rukovanje CORBA referencama svedeno na minimum (upravo ovaj segment zahteva posebnu pažnju prilikom implementacije, i smatra se jednim od najčešćih izvora grešaka). Međutim, nedostatak koji ne ide u prilog ovog modela svakako je njegova složena struktura. Dijamant nasleđivanje smatra se jednim od najkomplikovanijih koncepata objektnog programiranja, i preporuka je da se izbegava kad god je to moguće. Jedna od posledica koje otežavaju implementaciju, a samim tim i nadogradnju ovog univerzalnog modela, jeste i potreba za dinamičkim pretvaranjem tipova (*dynamic cast*).

Napomene:
a) Rad je proistekao iz diplomskog-master rada Ivane Popović. Mentor je bio prof.dr Branislav Atlagić.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

4. MODEL M2

Model M2 nastao je kao rezultat potrebe da se dijamant struktura modela M1 izmeni. Iz tog razloga, osnovna razlika ovog univerzalnog modela u odnosu na prethodni (M1) jeste način projektovanja test klasa za različite tipove sesija. U ovom rešenju, svaka test klasa formira se isključivo kao nadskup funkcionalnosti u odnosu na test klase za sesije od kojih je posmatrana sesija izvedena (sve test klase su disjunktne). Kao posledica toga, za testiranje određenog tipa sesije potrebno je instancirati više test objekata, a samim tim potrebno je voditi računa i o više CORBA referenci. Pojednostavljeni prikaz klasa klijent aplikacije u koju je ugrađen univerzalni model M2 prikazan je na slici 2.

U univerzalnom modelu M2, test klase za određeni tip sesije kreiraju se izvođenjem iz *CSessionHelper* klase, i čuvaju se u kontejneru sa sesijama. Kontejnerska klasa i sva funkcionalnost vezana za interaktivni rad deo je univerzalnog modela.



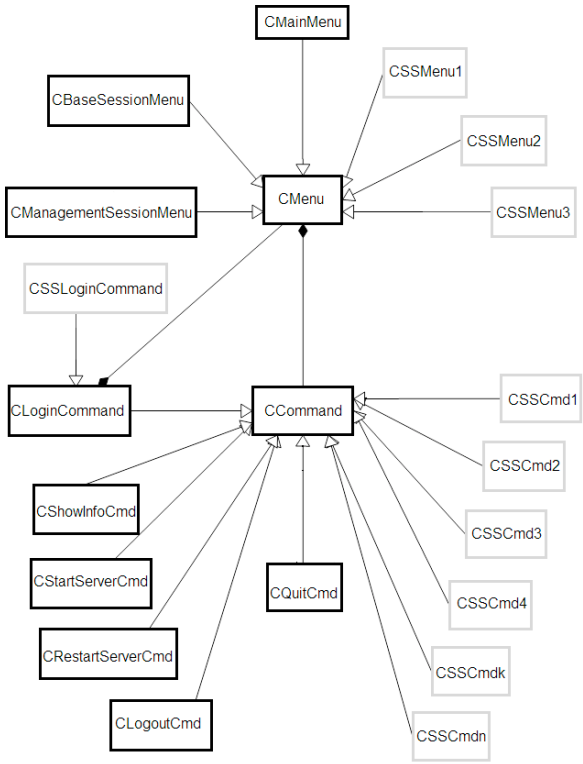
Slika 2. *Šematski prikaz klijentske aplikacije sa ugrađenim univerzalnim modelom M2*

Uočava se da je objektna struktura modela prilično jednostavna, tako da se može reći da su kriterijumi jednostavnosti upotrebe univerzalnog modela M2 i njegove proširivosti u velikoj meri zadovoljeni.

5. MODEL M3

Model M3 zasniva se na *Command* dizajn šablonu. U tom kontekstu, svaka klijentska test aplikacija posmatra se kao skup menija (koji omogućavaju interaktivni rad) i njima pridruženih komandi (čija je uloga izvršavanje akcije odabrane od strane korisnika). Organizacija menija u okviru aplikacije odslikava hijerarhiju CORBA sprega. Prikaz klasa klijent aplikacije u koju je ugrađen univerzalni model M3 dat je na slici 3.

Osnova ovog modela jesu klase *CMenu* i *CCommand*. Klasa *CMenu* predstavlja kontejnersku klasu za *CCommand* objekte. Svi meniji u okviru klijentske test aplikacije izvode se iz *CMenu* klase. Slično, sve komande u okviru aplikacije izvode se iz klase *CCommand*, i svaka od njih korisniku omogućava izvršavanje jedne akcije (akcija može biti slanje zahteva ka serveru ili otvaranje nekog od podmenija). Sva funkcionalnost neophodna za interaktivni rad deo je univerzalnog modela, kao i logika za koordinaciju menija i komandi. Nadogradnja za konkretnu klijentsku test aplikaciju zahteva dodavanje menija (i njima pridruženih komandi) za testiranje funkcionalnosti sesija specifičnih za server, i konkretne komande za prijavljivanje (*login*) korisnika. Rukovanje CORBA referencama slično je kao i kod modela M2. Može se reći da je ovaj model na zadovoljavajući način ispunio postavljene zahteve.



Slika 3. *Šematski prikaz klijentske aplikacije sa ugrađenim univerzalnim modelom M3*

6. UPOREDNA ANALIZA MODELA M1, M2 I M3 POMOĆU SOFTVERSKIH METRIKA

Za potrebe uporedne analize tri navedena modela korišćen je Resource Standard Metrics (RSM) programski paket, koji podržava različite vrste kvalitativne analize koda za programske jezike C, C++ i Java (nezavisno od operativnog sistema). Podržana je analiza broja linija koda, analiza logičke kompleksnosti, metrika objekata i nasleđivanja, funkcionalna i metrika modula. Takođe, moguće je meriti promene u kodu tokom vremena.

Prva kategorija softverskih metrika korišćenih za evaluaciju univerzalnih modela su metrike kvaliteta izvedbe koda. Ovaj tip uporedne analize nije od velikog značaja za ovaj rad, obzirom na to da je primarni cilj utvrđivanje razlika u objektnim strukturama modela M1, M2 i M3. Ipak, rezultati dobijeni ovom vrstom ocenjivanja značajno su uticali na poboljšanje kvaliteta samog koda univerzalnih klijentskih aplikacija. Iz ove kategorije metrika na univerzalne modele

primenjene su *Verbose Metrics* i *Total Quality Profile*. Navedeni su neki od indikatora koji se dobijaju pomoću metrika ove vrste:

- analiza broja linija koda: ovaj tip analize pojavljuje se u mnogim metrikama i može davati različite indikacije (neke od njih biće objašnjene kasnije)
- broj pojava ključnih reči (kao što su do, while, case, break, switch, new, delete...)
- analiza pojava različitih vrsta zagrada, space karaktera, praznih linija i komentara u kodu. Ovu podvrstu možemo okarakterisati kao parametar čitljivosti koda

Prilikom uporednog testiranja modela posmatranih u ovom radu akcenat je stavljen na analizu pomoću kategorije metrika koja daje ocenu kompleksnosti. Korišćene su *Class/Object Metrics Analysis* i *Complexity Metrics Detail, Cyclomatic, Interface, Total*. Obzirom na to da je primarni cilj utvrđivanje razlika u objektnim modelima, u nastavku će biti prikazani i objašnjeni neki od parametara dobijeni pomoću prve navedene metrike.

Class/Object Metrics Analysis generiše ocene preko 100 različitih parametara kako na nivou celog projekta, tako i za svaku pojedinačnu klasu. U tabeli 1 dati su delimični rezultati uporedne analize modula koji čine same univerzalne modele M1, M2 i M3. U tabeli 2 analizirani su posebno samo oni moduli koji predstavljaju nadogradnju specifičnu za konkretnu klijentsku aplikaciju (ideja je bila dobijanje procene kompleksnosti same nadogradnje, obzirom da je jednostavnost upotrebe možda i najvažniji zahtev koji se postavlja pred univerzalni model). U nastavku, navedena je selekcija parametra za koje se smatra da su od najvećeg značaja za uporednu analizu, kao i njihov kratak opis:

- ukupan broj klasa i metoda u modelu: predstavlja stepen objektne orjentisanosti sistema
- maksimalna dubina stabla nasleđivanja: predstavlja broj baznih klasa koje potencijalno mogu uticati na izvedenu klasu. Suštinski, što je veća dubina nasleđivanja, klasa se smatra kompleksnijom
- odnos baznih i izvedenih klasa: veća vrednost ovog parametra ukazuje na bolju ponovnu upotrebljivost koda (*code reusability*)
- broj fizičkih linija koda: predstavlja indikator veličine programa. Standardna preporuka je da svaka klasa u aplikaciji ima manje od 1000 linija koda
- *cyclomatic complexity*: broj linearno nezavisnih putanja kroz programski kod. Opisuje složenost toka programa
- kompleksnost sprege (*interface complexity*): ukupan broj parametara i tačaka povratka iz funkcije. Preporuka je da se nikada ne prosleđuje više od 6 parametara (pogrešan redosled prosleđivanja smatra se jednim od najčešćih izvora grešaka), kao i da uvek postoji tačno jedna tačka povratka iz funkcije

Parameter	M1	M2	M3
Total classes	5	5	17
Total methods	55	58	95
Max inheritance depth	1	1	1
Derived/Base ratio	0.5	0.67	4.67
Total pLOC	743	904	1258
Max pLOC	258	401	293
Avg pLOC	148.6	180.8	74
Total cyclomatic complexity	83	104	145
Max cyclomatic complexity	30	43	27
Avg cyclomatic complexity	16.6	20.8	8.53
Total interface complexity	87	70	112
Max interface complexity	26	23	14
Avg interface complexity	17.4	14	6.59

Tabela 1. *Analiza M1, M2 i M3 modela pomoću Class/Object Metrics Analysis*

Parameter	SS_M1	SS_M2	SS_M3
Total classes	4	3	20
Total methods	43	45	79
Max inheritance depth	2	1	1
Derived/Base ratio	1	1.5	6.67
Total pLOC	1519	1201	1580
Max pLOC	1051	1018	182
Avg pLOC	379.75	450.33	79.25
Total cyclomatic complexity	129	112	119
Max cyclomatic complexity	79	78	11
Avg cyclomatic complexity	32.25	37.33	5.95
Total interface complexity	72	71	91
Max interface complexity	38	43	10
Avg interface complexity	18	17.4	4.55

Tabela 2. *Analiza SS_M1, SS_M2 i SS_M3 modula pomoću Class/Object Metrics Analysis*

7. ZAKLJUČAK

Iz tabela 1. i 2. uočavaju se značajne razlike među modelima M1, M2 i M3. Kao prvo, velika je razlika u broju klasa i metoda modela M3, u odnosu na modele M1 i M2. Takođe, kod M3 modela (kao i kod specifične nadogradnje na model M3), vrednost parametra koji označava odnos broja baznih i izvedenih klasa (*derived/base ratio*) znatno je veća. Stoga, možemo zaključiti da model M3 ima veći stepen objektne orjentisanosti, i omogućava bolju iskorišćenost postojećeg koda.

Kod analize broja linija koda univerzalnih modela i odgovarajućih nadogradnji posebno se ističu maksimalne i prosečne vrednosti na nivou klasa. Uočava se da maksimalne vrednosti kod modula SS_M1 i SS_M2 prelaze vrednost od 1000, što je u direktnoj suprotnosti sa standardnom preporukom. Klase modula SS_M3 znatno su manje (za čak do 80%).

Analiza kompleksnosti sprege (*interface complexity*) i *cyclomatic complexity* parametra pokazuju da su ukupne (*total*) vrednosti veće za oko 25 – 30% kod modela M3 i SS_M3 modula (ovo je posledica znatno većeg broja klasa i metoda). Međutim, prosečne i maksimalne vrednosti višestruko su manje i kod modela M3 i kod njegove nadogradnje.

Iz navedenog, lako se izvodi zaključak da model M3 predstavlja skup većeg broja manjih i jednostavnijih klasa, dok su modeli M1 i M2 i njihove nadogradnje realizovani manjim brojem većih i kompleksnijih klasa. Dekompozicija velikih kompleksnih struktura na manje i jednostavnije celine smatra se dobrom praksom u objektnom programiranju.

Jednostavniji i manji moduli po pravilu su manje podložni greškama. Takođe, primena postojećih dizajn šablona znatno olakšava razumevanje, održavanje i nadogradnju.

Činjenica da model M3 predstavlja svaku komandu servera pojedinačnom klasom, uz jasno očuvanje logičkih celina – sesija, koje su modelovane odgovarajućim menijima, obezbeđuje maksimalnu fleksibilnost modela u slučaju promena u samim serverima.

Iako nijedna od navedenih metrika nije analizirala kompleksnost rukovanja CORBA referencama (što je veoma značajna stavka pri analizi univerzalnih modela ovakve namene), može se reći da model M3 predstavlja najbolje objekno orjentisano rešenje za univerzalnu, nadogradivu CORBA test klijentsku aplikaciju.

LITERATURA

U ovom radu korišćena je sledeća literatura:

[1] Duncan Grisby, "The omniORB version 4.0 User's guide" AT&T Laboratories, Cambridge.

[2] Bruce Eckel, „Thinking in C++", PlanetPDF, 2000

[3] Erich Gamma, Richard Helm, Ralph Johnson, John Vilssides "Design patterns – Elements Of Object Oriented Software", Addison Wesley Longman, Inc.

Abstract – This paper describes implementation process and comparative analysis for 3 different object oriented models for universal, reusable CORBA client application for testing server components contained in DMS softvare product. Most important criterias, evaluated by using softvare metrics tools, are simplicity of use and expandability.

DESIGNING, IMPLEMENTATION AND COMPARATIVE ANALYSIS OF DIFFERENT OBJECT ORIENTED CORBA CLIENT APPLICATIONS

Ivana Popović, Vladimir Savić, Petko Milidragović

JEDNO REŠENJE PROGRAMSKE PODRŠKE PROJEKTORA ZVUKA U DIGITALNOM TV UREĐAJU

Teodora Petrović, Dragan Simić, Tatjana Samardžić, Željko Lukač, Bogdan Trivunović

***Sadržaj* — U radu je prikazano jedno rešenje proširenja programske podrške za TV prijemnik sa implementacijom projektora zvuka. Date su osnovne karakteristike sistema, kratak opis realizacije i ispitivanja na fizičkoj platformi.**

***Ključne reči* — Projektor zvuka, dodatni procesor za obradu zvuka, digitalna televizija.**

I. UVOD

DA bi ostali konkurentni na tržištu proizvođači televizora moraju konstantno poboljšavati svoje proizvode usavršavanjem kvaliteta slike i zvuka, kao i istovremenim poboljšanjem dizajna televizora u cilju što veće jednostavnosti i lakšeg korišćenja.

Za postizanje više-kanalnog zvučnog okruženja u kućnim uslovima potrebno je obezbediti pet ili više zvučnika, raspoređenih na odgovarajućim mestima da bi se postigli najbolji rezultati, uključujući i mnoštvo kablova. Svaki put kada se obavlja razmeštaj u prostoriji, potrebno je ponovo postaviti zvučnike na odgovarajuća mesta.

Projektor zvuka eliminiše ove probleme i omogućava laku instalaciju i ponovno podešavanje. Ova tehnologija omogućava korisnicima više-kanalno zvučno okruženje korišćenjem samo jednog niza zvučnika smeštenih u digitalnom televizoru. Projektor zvuka emituje snopove zvuka koji se odbijaju o zidove prostorije i na taj način stvara više-kanalno zvučno okruženje.

Na Sl. 1. su prikazani snopovi koje emituje projektor zvuka, način na koji su raspoređeni i kako odbijanjem o zidove prostorije stižu do slušaoca. Levi, desni i centralni kanal stižu do slušaoca sa prednje strane, dok levi i desni zadnji kanali stižu od nazad tako da slušalac može osetiti više-kanalno zvučno okruženje kao i sa korišćenjem više zvučnika.

Ovaj rad je delimično finansiran od Ministarstva za nauku Republike Srbije, projekat 12004, od 2008 god.

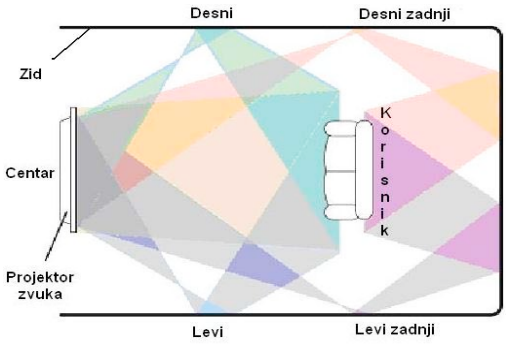
Teodora Petrović, Autor, Fakultet Tehničkih Nauka, Novi Sad, Srbija; (e-mail: teodora.petrovic@micronasnit.com).

Dragan Simić, Koautor, Fakultet Tehničkih Nauka, Novi Sad, Srbija; (e-mail: dragan.simic@micronasnit.com).

Tatjana Samardžić, Koautor, MicronasNIT, Institut za Informacione Tehnologije, Novi Sad, Srbija; (e-mail: tatjana.samardzic@micronasnit.com).

Željko Lukač, Koautor, MicronasNIT, Institut za informacione Tehnologije, Novi Sad, Srbijač (e-mail: zeljko.lukac@micronasnit.com).

Bogdan Trivunović, Koautor, MicronasNIT, Institut za informacione Tehnologije, Novi Sad, Srbija (e-mail: bogdan.trivunovic@micronasnit.com).



Sl. 1. Princip rada projektora zvuka.

II. ANALIZA PROBLEMA

Cilj projekta je implementacija programske podrške za projektor zvuka i njena integracija u postojeću aplikaciju za digitalni TV uređaj. Programska podrška treba da obezbedi funkcionalan sistem koji omogućava korišćenje svih režima rada projektora zvuka.

U okviru projekta implementirana je podrška za:

- Podešavanje režima rada projektora zvuka
- Automatsko podešavanje
- Ručno podešavanje svakog kanala
- Reinicijalizacija algoritma

A. *Podešavanje režima rada projektora zvuka*

Mogući su sledeći režimi rada projektora zvuka:

- 5 kanala – početni režim rada projektora zvuka koji ima pet odvojenih kanala (levi, desni, centar, levi zadnji i desni zadnji).
- 3 kanala – Ovaj režim koristi tri odvojena kanala (levi, desni i centar), dok su zadnji kanali izmešani sa odgovarajućim levim, odnosno desnim kanalom.
- 2 kanala (stereo) – Ovaj režim se koristi za reprodukciju običnog stereo zvuka. Koriste se samo dva kanala (levi i desni), dok su ostali kanali izmešani sa ova dva.

B. *Automatsko podešavanje projektora zvuka*

Da bi se automatski podesili parametri projektora zvuka, potrebno je uključiti mikrofon i postaviti ga na mesto gde sedi korisnik. Takođe je potrebno da jačina zvuka korišćenog pri automatskom podešavanju bude podešena u zavisnosti od udaljenosti mikrofona. Mikrofon mora biti postavljen na manje od 20º od centra projektora zvuka. Ukoliko su svi uslovi za automatsko podešavanje ispunjeni, projektor zvuka bi trebao da u roku od 16 sekundi podesi parametre za svaki kanal. U protivnom

Napomene:
a) Rad je proistekao iz diplomskog-master rada Teodore Petrović. Mentor je bio prof.dr Nikola Teslić.
b) Rad je prethodno publikovan na konferenciji TELFOR, Beograd, Novembar 2008.

sistem će prijaviti grešku. U toku ovog procesa, projektor zvuka emituje kratke zvučne signale u različitim pravcima koji dolaze do mikrofona i na taj način se određuju parametri za najbolje zvučno okruženje za tu prostoriju.

C. *Ručno podešavanje svakog kanala*

Ručno podešavanje kanala omogućava korisniku da podesi parametre za svaki kanal po svojoj želji. Moguće je podešavati samo one kanale koje trenutno koristi projektor zvuka, pa je prethodno potrebno podesiti broj kanala (5,3 ili 2 kanala). Korisniku je potrebno obezbediti podešavanje sledećih parametara za željeni kanal:

- Usmerenje snopa (direktno ili na zid) – podešava se samo za levi i desni kanal .
- Jačina zvuka kanala – raspon [-5 dB, +5 dB]
- Ugao snopa – raspon [-70º, +70º]
- Fokus snopa – raspon [-5m, +5m]

Kako bi se korisniku olakšalo ručno podešavanje parametara uključuje se beli šum za kanal koji se podešava.

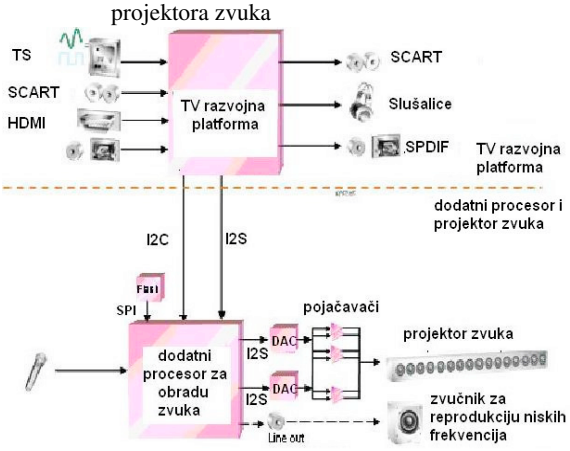
D. *Reinicijalizacija algoritma*

Reinicijalizacija algoritma treba da omogući postavljanje parametara za sve kanale na početne vrednosti.

III. FIZIČKA ARHITEKTURA SISTEMA

Rešenje obezbeđuje jednostavnu spregu korisnika sa aplikacijom koji će korišćenjem daljinskog upravljača kontrolisati rad projektora zvuka kao deo funkcionalnosti TV uređaja. Fizičku arhitekturu sistema čine:

- TV razvojna platforma – sa integrisanim kolom za digitalni TV uređaj,
- Dodatna ploča sa multistandardnim procesorom za obradu zvuka, koja obezbeđuje funkcionalnost projektora zvuka,
- Niz od 16 zvučnika – projektor zvuka,
- Fizička sprega između TV razvojne platforme i projektora zvuka



Sl. 2. TV razvojna platforma sa projektorom zvuka.

Kao TV platforma u ovoj implementaciji korišćena je razvojna platforma za digitalnu televiziju proizvođača Micronas. Ova namenska platforma zadovoljava sve zahteve za digitalne televizore; podržava višestruke TS ulaze i izlaze, analogne RGB, CVBS, Y/C izlaze, digitalne audio ulaze i izlaze S/PDIF i I2C.

Na dodatnoj ploči se nalazi multistandardni procesor za obradu zvuka proizvođača Micronas. Zvuk sa TV platforme se dovodi na procesor za obradu zvuka gde se obrađuje i šalje na DA pretvarače gde se pretvara u analogni signal i šalje na pojačavače i projektor zvuka.

Sprega između TV razvojne ploče i multistandardnog procesora za obradu zvuka ostvarena je pomoću:

- I²C sprege (Inter Integrated Circuit Bus) - za kontrolu.
- I²S sprege (Inter-IC Sound) - za prenos zvuka u digitalnom obliku.

IV. RAZVOJ PROGRAMSKE PODRŠKE

Programsku podršku čini proširena postojeća programska podrška za TV razvojnu platformu.

Realizovan je mehanizam koji dozvoljava jednostavno dodavanje i uklanjanje novih funkcija i modula neophodnih za realizaciju aplikacije.

A. *Razvojno okruženje*

Kao polazna osnova za razvoj programske podrške korišćena je postojeća TV razvojna aplikacija za TV mikrokontrolere. Razvojna aplikacija se prevodi i uvezuje pomoću KEIL prevodioca. Platforma je opremljena većim brojem audio/video ulazno/izlaznih priključaka. Platforma je putem standardne serijske veze RS 232 spojena sa personalnim računarom na kojem se nalazi i Starter alat. Starter se, u razvoju programske podrške za TV uređaj, koristi kao alat za praćenje toka izvršavanja programa.

Rad projektora zvuka kontroliše 8051 mikrokontroler. TV razvojna aplikacija ima odvojene nivoe koji vode računa o različitim funkcionalnostima. Na Sl. 3. prikazani su nivoi postojeće TV razvojne aplikacije koje je potrebno izmeniti za potrebe projektora zvuka.

Sl. 3. Arhitektura softvera.

TV programska podrška se sastoji iz sledećih nivoa:

- Aplikacioni nivo – Navigator (za menjanje kanala), TTX (teletekst), Menu System, itd.
- Međunivo – modul zadužen za postavljanje i promenu željene funkcionalnosti.
- Adaptacioni nivo – predstavlja centralni nivo i zadužen je za komunikaciju sa nivoom rukovaoca. Obezbeđuje da viši nivoi nemaju uvid u fizičku arhitekturu.
- Nivo rukovaoca – modul zadužen za direktnu spregu adaptacionog nivoa sa fizičkom podrškom.

V. OPIS REALIZACIJE

Programsko rešenje aplikacije za projektor zvuka obuhvata sledeće:

- Spoljni rukovalac za projektor zvuka (nalazi se na nivou rukovaoca i adaptacionom nivou).
- Izmene u postojećem rukovaocu za zvuk (na nivou rukovaoca).
- Izmene u međunivou za povezivanje aplikacionog i adaptacionog nivoa tj. za pozive iz korisničkog menija.

A. Rukovalac projektora zvuka

Rukovalac projektor zvuka je spoljni rukovalac koji se oslanja na najniži nivo programske podrške i koristi postojeće funkcije za pristup registrima multistandardnog procesora za obradu zvuka. Ovaj dodatni procesor ima specijalne registre koji se nalaze u memoriji.

Na nivou rukovaoca nalaze se moduli za pristup registrima uz prethodnu proveru ispravnosti opsega adresa registara dodatnog procesora.

U modulu *sp_reg* su implementirane funkcije za očitavanje i upis u registre dodatnog procesora. Funkcije se oslanjaju na već postojeće funkcije za pristup registrima.

Parametri (ugao i fokus) svakog kanala i jačina zvuka automatskog podešavanja imaju svoje registre, dok se usmerenje, pokretanje automatskog podešavanja, statusi o greškama, itd. nalaze u posebnom kontrolnom registru (kontrolni-status registar). Jačina zvuka kanala se ne podešava u rukovaocu projektora zvuka već podešavanjem postojeće jačine zvuka za određeni kanal.

Čitanje i pisanje registara preko I²C komandi obavlja se na sledeći način:

Sl. 4. Pristup registrima dodatnog procesora.

Za pristup registrima dodatnog procesora za obradu zvuka koristi se identifikator dodatnog procesora (0xID), pod-adresa (0xPA) i adresa registra (0xRA). Prilikom pisanja potrebno je poslati i vrednost koja se upisuje (0xDD).

U kontrolnom registru podešavaju se sledeće funkcije i parametri za projektor zvuka:

- reinicijalizacija – omogućava postavljanje parametara svih kanala na početne vrednosti.
- levo direktno usmerenje – omogućava da se levi kanal ponaša kao kod standardnih zvučnika. U ovom slučaju, levi kanal se emituje iz poslednjih 6 zvučnika u nizu sa leve strane. Kada je uključeno direktno usmerenje, korisnik nema mogućnost podešavanja ugla i fokusa levog kanala i umesto njih se koriste ugao i fokus centralnog kanala.
- desno direktno usmerenje - omogućava da se desni kanal ponaša kao kod standardnih zvučnika. U ovom slučaju, desni kanal se emituje iz poslednjih 6 zvučnika u nizu sa desne strane. Kada je uključeno direktno usmerenje, korisnik nema mogućnost podešavanja ugla i fokusa desnog kanala i umesto njih se koriste ugao i fokus centralnog kanala.
- automatsko podešavanje – za pokretanje automatskog podešavanja projektora zvuka kao i za proveru da li je proces završen.
- indikator greške u slučaju da mikrofon nije spojen ili je signal slab.
- indikator greške da je centralni ugao van opsega – mikrofon ne sme biti postavljen na više od 20º od centra projektora zvuka.

Na adaptacionom nivou se nalaze moduli u kojima je implementirana kompletna funkcionalnost vezana za projektor zvuka.

Dodat je novi modul *a_aud_ctrl_sp* (u kome je implementiran rukovalac projektora zvuka) i sprovedene su izmene u postojećem *a_aud_ctrl* (u kome je implementirana funkcionalnost vezana za zvuk na adaptacionom novou).

U modulu *a_aud_ctrl_sp* su implementirane funkcije za ručno podešavanje ugla, fokusa i usmerenja željenog kanala, kao i za pokretanje i proveru da li je završeno automatsko podešavanje projektora zvuka i podešavanje jačine zvuka za vreme trajanja tog procesa.

Funkcija za reinicijalizaciju projektora zvuka takođe je realizovana u ovom modulu.

Sve funkcije u ovom modulu oslanjaju se na prethodno opisane module na nivou rukovaoca. Sa adaptacionog nivoa prosleđuju se registar i vrednost koja treba da se upiše, kao i identifikator dodatnog procesora za obradu zvuka.

U modulu *a_aud_ctrl* dodate su funkcije za podešavanje jačine zvuka za više kanala kao i za uključivanje belog šuma za željeni kanal.

Za realizaciju više-kanalnog zvučnog okruženja neophodno je omogućiti podešavanje jačine zvuka za svaki kanal. Takođe se prilikom ručnog podešavanja jačine zvuka koristi ista funkcija ali se poziva za određeni kanal.

Beli šum se uključuje za svaki kanal posebno prilikom ulaska u meni za taj kanal i isključuje nakon izlaska iz menija. Šum se generiše u jednom od internih blokova dodatnog procesora za obradu zvuka.

B. Izmene u postojećem rukovaocu zvuka

Standardni televizor koristi dva kanala – levi i desni, dok projektor zvuka reprodukuje pet odvojenih kanala, pa je potrebno izmeniti postojeće blokove za obradu zvuka. Zbog više-kanalnog okruženja, izmenjeni su interni blokovi procesora za obradu zvuka.

U jednom od internih blokova po potrebi se koristi „Dolby Pro Logic“ kako bi pretvorio stereo ulazni signal u pet odvojenih kanala tako da je neophodno da uvek bude podešen na taj režim rada osim kada je uključen šum i kada interni blok mora biti postavljen na generator šuma ili kada se koristi više-kanalni ulazni signal.

Za većinu internih blokova bilo je potrebno uvesti nove ili iskoristiti postojeće blokove kako bi se podržalo više-kanalno zvučno okruženje. Tako npr. svi blokovi korišćeni

za slušalice su isključeni da bi se iskoristili za novo više-kanalno okruženje.

C. *Izmene u međunivou projektora zvuka*

Međunivo komunicira direktno sa aplikacijom pa su implementirane funkcije koje povezuju aplikacioni i adaptacioni nivo.

U postojećem modulu *m_aud_ctrl* implementirane su funkcije za podešavanje svih parametara jednog kanala, izbor režima rada projektora zvuka (5,3 ili 2 kanala), kao i funkcije za pokretanje i proveru stanja procesa automatskog podešavanja projektora zvuka, koje se oslanjaju na funkcije na adaptacionom nivou. Prilikom izbora režima rada projektora zvuka, podešavani su interni blokovi postojećeg rukovaoca zvuka. Ukoliko je izabrani režim rada stereo– potrebno je da se spoje ostali kanali u levi i desni, dok se za slučajeve sa 3 i 5 kanala propuštaju svi kanali.

VI. ISPITIVANJE

Nakon prevođenja programske podrške i spuštanja na ciljnu platformu, ispitane su sve podržane funkcije u stvarnoj upotrebi, čime je potvrđen ispravan rad.

Za ispitivanje je korišćena „Visual I²C“ aplikacija pomoću koje se pristupalo memoriji i registrima integrisanog kola za digitalni TV i multistandardnog procesora za obradu zvuka preko I²C serijske sprege i proveravala valjanost podataka sa unapred definisanih adresa.

Sl. 5. Izgled menija projektora zvuka.

Sl. 6. Izgled menija za ručno podešavanje levog kanala.

Sl. 7. Automatsko podešavanje projektora zvuka.

Ispitivanje je obavljeno i interaktivnom simulacijom akcija korisnika u okviru menija. Ispitane su sve predviđene situacije i ponašanje sistema u njima.

Na slikama 5, 6 i 7 prikazana je korisnička aplikacija tj. izgled glavnog menija projektora zvuka, menija za ručno podešavanje levog kanala i za automatsko podešavanje projektora zvuka.

VII. ZAKLJUČAK

Ovaj rad je primer kako se funkcije digitalnog TV uređaja mogu proširiti nestandardnim mogućnostima kao što je projektor zvuka. Ovaj sistem predstavlja dobru polaznu osnovu za dalji razvoj i proširenje.

Postoji mogućnost proširenja postojeće aplikacije dodavanjem novih funkcija kao što su specijalni zvučni efekti, uvođenje novih režima rada ili poboljšanje postojećih funkcija.

Ovaj model rešenja je lako primenjiv i na druge TV platforme i slične sisteme.

ZAHVALNICA

Ovaj rad je delimično finansiran od Ministarstva za nauku Republike Srbije, projekat 12004, od 2008. god.

LITERATURA

[1] Programska podrška u televiziji i obradi slike, Dr Nikola Teslić, materijali za predavanja, FTN Novi Sad, 2005.
[2] SoundProjectorSystemSpec.pdf, Micronas GmbH, Freiburg, April 2008.
[3] SoundProjectorV1UserManual v1_10.pdf, Micronas GmbH, Freiburg, April 2008.
[4] MSP_M_MAP_M_D4_Audio_Processor_Family.pdf, Micronas GmbH, Freiburg, January 2008.

ABSTRACT

This paper presents one implementation of sound projector for digital TV device. Digital Sound Projector produces beams of sound that are projected into the room and reflected from walls to create surround sound. The technology provides a complete surround sound solution, which replaces an amplifier, a lot of cables and multiple speaker boxes.

ONE IMPLEMENTATION OF SOUND PROJECTOR FOR DIGITAL TV DEVICE

Teodora Petrović, Dragan Simić, Tatjana Samardžić, Željko Lukač, Bogdan Trivunović

Automatizovano simulaciono okruženje za ispitivanje inteligentnih senzora zasnovanih na MEMS

Aleksandar Tucakov, Nebojša Pjevalica, Zoltan Pele

***Sadržaj* — Rad prikazuje jedan pristup u projektovanju i korišćenju automatizovanog okruženja za ispitivanje savremenih inteligentnih senzora zasnovanih na mikroelektromehaničkim (MEMS) davačima, koji poseduju digitalni sprežni modul.**

***Ključne reči* — Automatizacija ispitivanja, ispitivanje integrisanih kola, MEMS, verifikacija.**

I. UVOD

ISPITIVANJE spada među najznačajnije faze u procesu razvoja integrisanih kola. U toku ispitivanja proverava se funkcionalnost razvijenog modela integrisanog kola pre njegovog puštanja u komercijalnu proizvodnju. Za opis modela integrisanog kola koriste se jezici za opis fizičke arhitekture kao što su Verilog ili VHDL [1], [2], [3] dok se samo ispitivanje/simulacija obavlja specijalizovanim softverskim okruženjima kao što su Modelsim i NCSim.

Blagovremeno ispitivanje pojedinih razvijenih modula (podblokova) povećava konačnu pouzdanost integrisanog kola. Nakon verifikacije svih pojedinačnih modula, neophodno je objediniti ih u celinu koja mora odgovarati željenom integrisanom kolu. Ovako formiran objedinjen model pruža mogućnost verodostojne provere sistema pre ulaska u proces serijske proizvodnje, nakon koga naknadne ispravke više nisu moguće.

Automatizacija procesa ispitivanja ima mnoge prednosti u odnosu na ručno ispitivanje. Pre svega ljudsko vreme je skupo a takođe, veoma je teško zadržati koncentraciju u dužem vremenskom periodu, pa je i verovatnoća da će čovek pogrešiti veća. Iako je za projektovanje automatizovanog okruženja u početku potrebno izdvojiti malo više truda, taj trud kasnije rezultuje većom pouzdanošću, te uštedom vremena i novca.

Pri ispitivanju je korišćen HDL model koji predstavlja deo senzora bez MEMS-a. Za sintezu je korišćen Xilinx ISE 9.2 [4] a za simulaciju softverski paket NCSim [5] kompanije Cadence.

II. KONCEPT SENZORA ZASNOVANIH NA MEMS TEHNOLOGIJI

U mikroelektromehaničke sisteme (MEMS) spadaju uređaji dimenzija između 1μm i 1mm proizvedeni kombinovanjem mehaničkih i električnih struktura [6]. Proizvodna tehnologija najvećim delom preuzeta je od proizvođača integrisanih kola. Značajno smanjene dimenzije ovakvih uređaja opredeljuju njihove mehaničke osobine, izuzetno mala masa, smanjen moment inercije, nizak koeficijent trenja, smanjeno mehaničko habanje doprinose dugotrajnosti. Navedene karakteristike implicirale su vrlo širok spektar primena pogotovo u senzorici. Tipični predstavnici MEMS senzora su akcelerometri, žiroskopi, senzori pritiska, protoka, termometri, anemometri, itd.

U osnovi MEMS senzor pretvara mehaničku ili neku drugu fizičku veličinu u električnu, koja se digitalizuje i prilagođava korisniku (filtrira, reskalira, itd.). U konkretnom slučaju je analiziran i simuliran akcelerometar koji meri ubrzanje po sve tri ose Dekartovog koordinatnog sistema. Akcelerometar se sastoji od seizmičke mase koja pretvara ubrzanje u silu, čijim delovanjem seizmička masa menja svoj položaj [7]. Njeno ogibljenje je elastično tako da je pomeraj srazmeran sili i menja kapacitet kondenzatora kojeg formira pokretni deo senzora sa „stacionarnim polisilikonskim prstima". Ovakva struktura je predstavljena na Sl. 1.

Ovaj rad je delimično finansiran od Ministarstva za nauku Republike Srbije, projekat 11005, od 2008. god.

A. Tucakov , Fakultet tehničkih nauka , Novi Sad, Srbija, (telefon: +381-(0)21-4801-186; faks: +381-(0)21-4801-196; e-mail: aleksandar.tucakov@rt-rk.com)

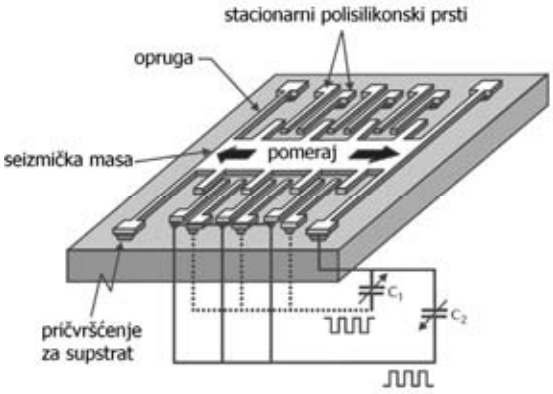
N. Pjevalica, Fakultet tehničkih nauka, Novi Sad, Srbija, (e-mail: nebojsa.pjevalica@rt-rk.com)

Z. Pele, Fakultet tehničkih nauka, Novi Sad, Srbija, (e-mail: zoltan.pele@rt-rk.com)

Napomene:

a) Rad je proistekao iz diplomskog-master rada Aleksandra Tucakova. Mentor je bio prof.dr Vladimir Kovačević.

b) Rad je prethodno publikovan na konferenciji TELFOR, Beograd, Novembar 2008.



Sl. 1. Akcelerometarski senzor zasnovan na pretvaranju ubrzanja u pomeraj.

Senzor u sebi sadrži tri ovakve strukture, orijentisane kolinearno sa svakom od osa koordinatnog sistema. Signali sa kondenzatora se vode na A/D pretvarač gde se pretvaraju u numeričke vrednosti srazmerne ubrzanju svake od osa. Analiza, simulacija i ispitivanje MEMS elementa izlazi iz okvira ovog rada. Realni podaci o ubrzanju preuzeti su sa fizičkog senzora i memorisani u ispitne datoteke. Ovako formiran skup ispitnih datoteka korišćen je pri ispitivanju digitalnog sklopa razvijanog senzora.

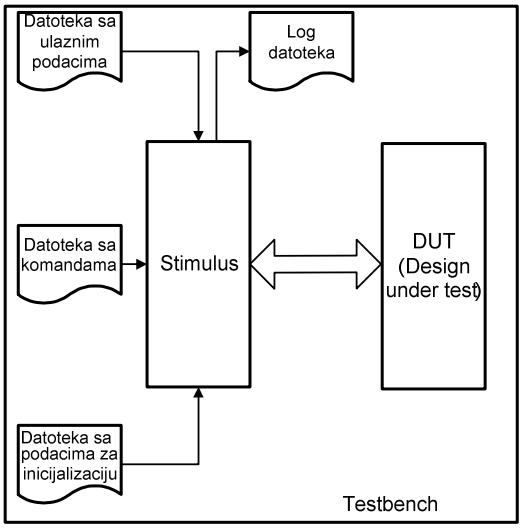
III. Opis ispitnog okruženja

Kao što je već napomenuto ispitivanje je vršeno nad VHDL dizajnom koji predstavlja deo senzora bez MEMS-a. Zbog nepostojanja adekvatnog modela MEMS-a realizovanog u VHDL-u javila se potreba da se na neki način simulira funkcionalnost samog MEMS-a. Ovo je učinjeno na taj način što je formirana datoteka sa ispitnim vektorima čije vrednosti predstavljaju vrednosti ubrzanja na izlazu samog MEMS-a. Na taj način iščitavanjem vrednosti iz datoteke i njihovim slanjem ka senzoru sa određenom frekvencijom simulira se funkcionalnost MEMS-a u celokupnom sistemu. Sva komunikacija prema senzoru obavlja se putem SPI (Serial Peripheral Interface) komunikacije.

Za ispitivanje je korišćen specijalizovan softverski paket namenjen za softversku simulaciju NCSim. Softverski paket NCSim sastoji se od sledecih komponenti:

- Simulator za Verilog jezik – NCVlog
- Simulator za VHDL jezik – NCVHDL
- Grafičko okruženje za rad sa NCVHDL i NCVlog – SimVision

Da bi se ispitao neki HDL model neophodno je pre svega napisati HDL kod koji će se ponašati kao "spoljašnje okruženje" za ispitivani HDL model. Ovakav HDL kod se na engleskom naziva Testbench [8] dok se modul koji se ispituje naziva DUT (Design under test). Uloga Testbench-a je da generše takt signale (clock), signale za dovođenje DUT-a u početno stanje (reset) i neophodne signale za pobuđivanje ispitivanog HDL modela. Isto tako Testbench posmatra izlazne signale ispitivanog HDL modela i u zavisnosti od njih generiše dodatne signale za pobuđivanje DUT-a ili ih beleži u jednu log datoteku kako bi korisnik kasnije bio u mogućnosti da ih pregleda.



Sl. 2. Izgled ispitnog okruženja

Na Sl. 2 je prikazan izgled ispitnog okruženja. Na samom početku neophodno je izvršiti inicijalizaciju samog modela koji se ispituje. Ovo se obavlja tako što se iz datoteke pročitaju podaci koji služe za inicijalizaciju i zatim se proslede ka modelu koji se ispituje. Nakon toga započinje se sa slanjem vrednosti ubrzanja ka modelu koji testiramo. Vrednosti ubrzanja predstavljaju izlaze MEMS-a koji postoji u realnom senzoru. U pitanju su osmobitne vrednosti ubrzanja predstavljene u komplementu dvojke. Ove vrednosti prosleđuju se ka modelu senzora određenom frekvencijom koju korisnik može unapred definisati pre početka ispitivanja.

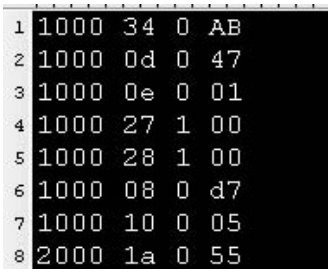
IV. Proces ispitivanja

Pri ispitivanju su korišćene tri ulazne tekstualne datoteke sa podacima. Prva datoteka sadrži podatke potrebne za inicijalizaciju modela koji se ispituje. Svaka linija ove datoteke predstavlja jedan podatak za inicijalizaciju i sastoji se iz adrese internog registra senzora u koji treba upisati vrednost i vrednosti koja se upisuje. Adrese i vrednosi prikazane su u heksadecimalnom formatu iz opsega 00h - FFh. Podaci iz ove datoteke prosleđuju se ka ispitivanom modelu na samom početku simulacije čime je sistem inicijalizovan i spreman za ispitivanje njegove funkcionalnosti.

Druga datoteka sadrži realne podatke o ubrzanju koji su preuzeti sa fizičkog senzora i zatim memorisani u datoteku. Uloga ove datoteke je da simulira funkcionalnost MEMS elementa tako što se iz nje iščitaju vrednosti koje se zatim

prosleđuju ka ispitivanom modelu. Frekvencija kojom se vrednosti ubrzanja prosleđuju ka ispitivanom modelu može se podesiti pre početka same simulacije ili se može koristiti podrazumevana vrednost. Za svaki ispitivani slučaj potrebno je formirati posebnu datoteku ovog tipa, u zavisnosti od toga koji modul (podblok) i koja funkcionalnost se želi ispitati.

Datoteka sa komandama sadrži komande koje su potrebne za ispitivanje senzora. Uloga ove datoteke je da Stimulusu obezbedi podatke neophodne za sukcesivno pobuđivanje modula (podblokova) ispitivanog modela. Nju kreira sam korisnik, u zavisnosti od toga koji modul i koje funkcionalnosti se žele ispitati. Svaka linija ove datoteke predstavlja jednu komandu za čitanje ili upis u željeni registar senzora. Izgled dela datoteke sa komandama je prikazan na Sl. 3.



Sl. 3. Izgled dela datoteke sa komandama

Linija datoteke sadrži trenutak u kojem treba izvršiti datu komandu, predstavljeno vremenskim intervalom od početka simulacije u mikrosekundama, adresu internog registra senzora (heksadecimalno) u koji treba upisati ili iz kojeg treba iščitati vrednost, indikator da li treba izvršiti upis (indikator je tada 0) ili čitanje (indikator je tada 1) i vrednost koju treba upisati (heksadecimalno). U slučaju komande za čitanje iz internog registra senzora vrednost u poslednjoj koloni komandne linije nije od interesa, međutim neophodno je upisati proizvoljnu vrednost iz opsega 00h – FFh zbog načina rada funkcije za čitanja iz datoteke.

Nakon formiranja ove tri vrste ulaznih datoteka i izbora izlaza koji se žele beležiti u izlaznu log datoteku započinje se sa procesom ispitivanja (simulacije). Kao rezultat simulacije koja traje određeni vremenski period tokom koga prisustvo čoveka nije neophodno dobijene su izlazne log datoteke, za svaki ispitivani slučaj po jedna, koje su zatim detaljno proučavane kako bi se izvršilo poređenje aktuelnog ponašanja digitalnog sistema sa željenim i kako bi se otkrile eventualne greške u digitalnom sistemu.

V. Rezultati ispitivanja

Detaljno ispitivanje ponašanja digitalnog senzorskog podbloka u okruženju opisanom u poglavlju III omogućilo je brzu lokalizaciju grešaka i njihovo ispravljanje. Za proveru svakog elementa digitalnog sklopa formirana je posebna datoteka sa pobudama. Poređenjem aktuelnog ponašanja sistema sa željenim, detekcija grešaka postaje efikasna i fokusirana.

Nakon ispravljanja uočenih grešaka (koncepcijskih i slučajnih) finalni dizajn je verifikovan u konačnom testu, gde su sukcesivno aktivirane sve pobude jedna za drugom uz praćenje ponašanja sistema.

VI. Zaključak

Imajući u vidu visoku cenu proizvodnje integrisanog kola, neotkrivene greške mogu proizvod učiniti neupotrebljivim na tržištu. Ovakav gubitak u oštroj tržišnoj konkurenciji, kakva je danas prisutna u svim oblastima informacione tehnologije, mogao bi biti nenadoknadiv.

Korak dalje u pristupu predprodukcijskom ispitivanju bio bi uključivanje MEMS elementa u simulacioni model. Ovakav sveobuhvatni pristup omogućio bi gotovo potpuno verodostojan proces verifikacije svih segmenata integrisanog kola u jedinstvenoj simulaciji. Time bi se dale ispraviti i eventualne greške u samom mikroelektromehaničkom bloku, kao i u sprezi između njega, A/D bloka i digitalnog bloka.

Literatura

[1] K. Skahill ,"VHDL for Programmable Logic" , Cypress Semiconductor, Addison-Wesley 1996, Menlo Park, CA
[2] P. Chu, "RTL Hardware Design Using VHDL", Wiley – Interscience, 2006
[3] S. Brown, Z. Vranesic, "Fundamentals of Digital Logic with VHDL Design", McGraw-Hill, 2000. Blacklick, OH
[4] Xilinx ISE 9.2 Software manuals, Xilinx Inc., 2007
[5] Cadence NC-VHDL Simulator help , 2004
[6] M. Gad-el-Hak, "MEMS Applications", 2nd ed, Boca Raton: CRC Taylor and Francis, 2006.
[7] N. Maluf, K. Williams, "An Introduction to Microelectromechanical Systems Engineering", Norwood, MA: Artech House, 2004.
[8] R. Munden , "ASIC and FPGA Verification: A Guide to Component Modeling", Morgan Kaufmann, 2004

Abstract

This paper presents single approach in design and usage of automated environment for MEMS based intelligent sensor testing. The target testing subblock is digital interface.

AUTOMATED SIMULATION ENVIRONMENT FOR INTELLIGENT MEMS BASED SENSORS TESTING

Aleksandar Tucakov, Nebojsa Pjevalica, Zoltan Pele

JEDNO REŠENJE IMPLEMENTACIJE ALGORITMA BRZE FURIJEOVE TRANSFORMACIJE NA 24-BITNOJ DSP PLATFORMI

Nemanja Popović, Marija Đekić, Mihajlo Katona i Boris Radin , *Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Katedra za računarsku tehniku i računarske komunikacije*

Sadržaj – *U ovom radu je prikazano jedno rešenje implementacije Rediks-2 algoritma brze Furijeove transformacije na 24-bitnom digitalnom signal procesoru aritmetike nepokretnog zareza sa korišćenjem dinamičke kontrole opsega. Takođe je opisano prilagođenje implementacije platformi kao i način realizacije ključnih segmenata.*

1. UVOD

Razvoj novih medija kao što su internet, DVD, digitalna televizija i slično su doveli do razvoja kompleksnih algoritama za kompresiju audio signala. Ti algoritmi su uglavnom zasnovani na spektralnim analizama visoke rezolucije i korišćenjem ljudskog psihoakustičnog modela.

Za računanje spektralne analize se koristi upravo Furijeova transformacija, koja ima široku primenu u digitalnoj obradi signala.

Brza Furijeova transformacija ili BFT (FFT – Fast Fourier Transformation) je algoritam za "brzo" izračunavanje vrednosti diskretne Furijeove transformacije (DTF).

U daljem tekstu biće opisan način implementacije jedne metode brze Furijeove transformacije na 24-bitnom digitalnom signal procesoru.

2. OPIS PROCESORA I PROBLEMA

Potrebno je implementirati algoritam BFT na 24-bitnom digitalnom signal procesoru. Tu se javljaju problemi tačnosti i brzine. Potrebno je izračunati BFT za najkraće moguće vreme a da greška računanja bude što je moguće manja.

Digitalni signal procesor se sastoji iz sledećih blokova: RAM-a, ROM-a, ALU jedinice i registara specijalne namene. Podržava celobrojnu i aritmetiku nepokretnog zareza.

RAM je organizovan u dve memorijske zone sa 24-bitnim memorijskim rečima.

Platforma ima mogućnost paralelne obrade stereo signala zato što ima dve ALU jedinice, gde svaka poseduje 24*12-bitni množač i 40-bitni akomulator za smeštanje rezultata. Rezultati se mogu smeštati direktno u RAM korišćenjem programibilnog Barelovog pomerača. Svaka jedinica ima indikator (fleg) prekoračenja ili potencijalnog prekoračenja opsega.

Standardno adresiranje je indirektno sa pomerajem (offset). Pomeraj može biti ili deo instrukcijskog koda ili zadan u registru pomeraja (offset register). Platforma poseduje pet pokazivača (pokazivačkih registara).

Takođe jedna od bitnih karakteristika platforme je ta da je procesor protočne (pipeline) strukture, što u mnogim situacijama otežava samo pisanje koda.

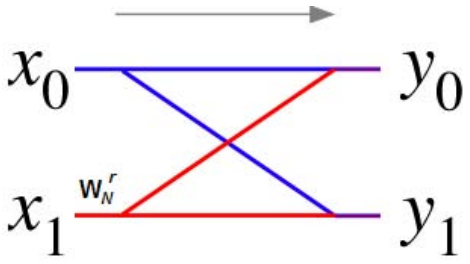
3. IMPLEMENTACIJA

Implementiran je Rediks-2 BFT algoritam sa decimacijom u vremenu (Decimation In Time - DIT).

Uočava se da su u prva dva stepena BFT vrednosti sinus-cosinus koeficijenata (tweedle factors) jedan ili nula. Ta činjenica veoma pojednostavljuje i ubrzava računanje izbacivanjem nepotrebnog množenja koje oduzima puno vremena (posle instrukcije množenja rezultat se dobija tek 6 instrukcija kasnije), tako da se operacije leptira (butterfly) svode na sabiranje i oduzimanje. Iz tog razloga algoritam je razložen na tri dela: **prvi** stepen, **drugi** stepen i **ostali** stepeni što je prikazano na dijagramu (*Slika2*).

U prvom stepenu je, radi ubrzanja, paralelno sa računanjem leptira urađeno i preuređenje ulaznih odbiraka bit-inverzijom (engl. bitreverse).

Grafički prikaz računanja **leptira** za DIT BFT je dat na sledećoj slici (*Slika1*) :



Slika 1*: Osnovni oblik leptira*

4. KONTROLA OPSEGA

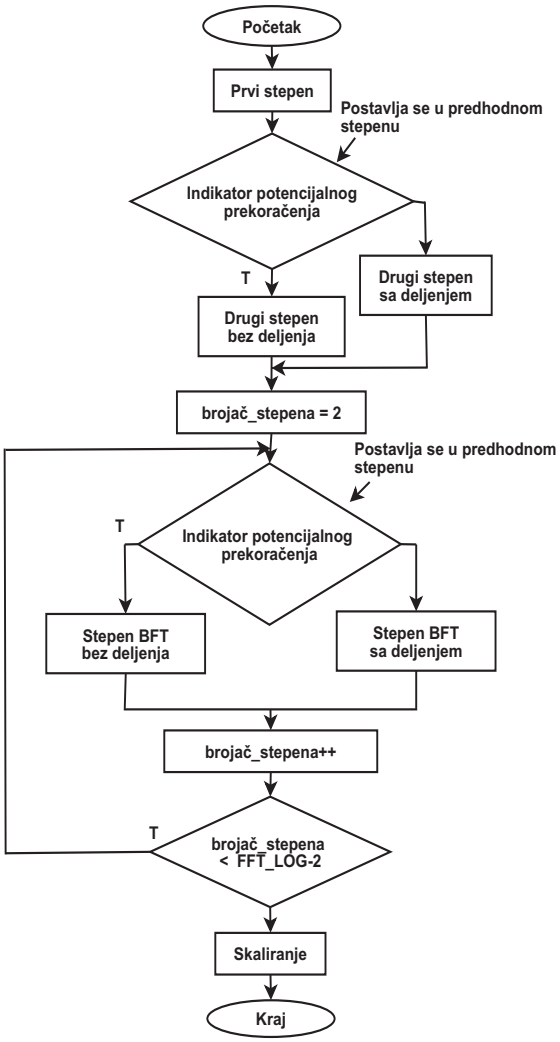
Usled ograničenja aritmetike nepokretnog zareza, konačne dužine memorijskih lokacija, registara i akumulatora, prilikom računanja dolazi do neizbežnog narušavanja tačnosti dobijenih rezultata. Kako bi se očuvala što je moguće veća tačnost rezultata, implementirana je i opciona **kontrola opsega** (headroom control). Ideja je da se prilikom računanja BFT u nekom od stepena obrade proverava indikator potencijalnog prekoračenja.

U zavisnosti od toga da li se u predhodnom stepenu obrade umesto nule u indikatoru pojavi jedinica, u sledećem stepenu će se dobijeni rezultati računanja deliti sa dva da prilikom operacija množenja i sabiranja ne bismo izašli van opsega. U suprotnom, u sledećem stepenu BFT neće biti deljenja. Na taj način se iskorištavaju svi biti memorijske reči odnosno registra za predstavu brojeva i rezultata računanja. Samim tim uz veće iskorišćenje opsega i rezultati su tačniji. Razdvajanje na stepene sa i bez deljenja je grafički prikazano na sledećem dijagramu (*Slika2*) :

Napomene:

a) Rad je proistekao iz diplomskog-master rada Nemanje Popovića. Mentor je bio prof.dr Branislav Atlagić.

b) Rad je prethodno publikovan na konferenciji ETRAN, Herceg Novi, Juni 2007.



Slika2: Blok dijagram BFT sa kontrolom tačnosti

5. SIMULACIJA ARITMETIKE DIGITALNOG SIGNAL PROCESORA

U programskom jeziku C je implementirana simulacija aritmetike nepokretnog zareza digitalnog signal procesora. Simulirani su 36-bitni akomulatori, 24-bitni registri i memorijske lokacije, kao i 12-bitne konstante.

Verno su simulirane aritmetičke operacije sabiranja, oduzimanja, množenja, pomeranja, kao i upisa u memoriju. Takođe, simulirani su i indikatori prekoračenja i potencijalnog prekoračenja.

Prvo je napisan algoritam BFT uz pomoć simulacije aritmetike posmatranog digitalnog signal procesora u C-u, i kada je postignuta željena tačnost prešlo se na implementaciju u asembleru. Za identične ulazne podatke porede se rezultati simulacije aritmetike i rezultati dobijeni u simulatoru procesora. Dobijeni su potpuno jednaki rezultati.

6. ISPITIVANJE

U tabeli 1 su dati rezultati ispitivanja za frekvencije 1000 Hz za levi i desni kanal, za sinusne signale punog opsega (Full Scale) :

Veličina prozora	**Apsolutni nivo šuma [dBFS]**	**Signal [dBFS]**
8	-200.00/-200.00	-0.01/-0.01
64	-135.46/-135.46	-3.01/-3.01
128	-135.46/-135.46	-3.01/-3.01
256	-135.46/-135.46	-3.01/-3.01
1024	-133.70/-133.70	-3.01/-3.01

Tabela1: Rezultati testiranja 1

U tabeli 2 su predstavljeni rezultati ispitivanja za frekvencije 1000 Hz za levi i desni kanal, višetonski (Multy Tone) signal punog opsega (Full Scale):

Veličina prozora	**Apsolutni nivo šuma [dBFS]**	**Signal [dBFS]**
8	-200.00/-200.00	-6.03/-9.04
64	-200.00/-200.00	-14.48/-14.48
128	-200.00/-200.00	-17.79/-17.79
256	-200.00/-200.00	-20.94/-20.94
1024	-200.00/-200.00	-27.08/-27.08

Tabela2 : Rezultati testiranja 2

Broj ciklusa za računanje BFT pri frekvenciji odabiranja 48000 Hz za različite veličine prozora BFT dat je u tabeli 3:

Veličina prozora	MIPS-a
8	2,44800
64	3,64800
128	4,14300
256	4,68225
1024	5,77950

Tabela3 : Broj MIPS-a za frekvenciju odabiranja 48 kHz

7. ZAKLJUČAK

U radu se pokazalo da je, iako relativno skromnih resursa, arhitektura DSP procesora vrlo pogodna za realizaciju raznih algoritama za digitalnu obradu signala.

Što se tiče poboljšanja tačnosti, pokazalo se da implementacija opcione kontrole opsega u znatnoj meri uvećava veličinu koda, ali ne utiče znatno na smanjenje brzine računanja BFT.

Napomena: Rad je delimično podržan u okviru projekta TR-6136B Ministarstva za nauku i zaštitu životne sredine Republike Srbije.

LITERATURA

[1] Stevan Berber, Miodrag Temerinac, *"Osnovi algoritama i struktura DSP,"* FTN Izdavaštvo, Novi Sad, 2004.

[2] Z. Lukac and M. Temerinac, "*One method for maintaining accuracy in implementation of Fast Fourier Transform on Fixed Point Digital Signal Processors*", MIPRO 2006, XXIX. International Convention, Opatija

Abstract – This paper deals with one solution of implementation of Radix-2 in-place FFT algorithm on 24-bit DSP procesor with implementation of headroom control. It is described how the algorithm is adapted to the architecture of platform, and how are implemented essential segments.

ONE SOLUTION OF IMPLEMENTATION OF FFT ALGORITHM ON 24-BIT DSP PROCESOR

Nemanja Popović, Marija Đekić,
Mihajlo Katona and Boris Radin

JEDNO REŠENJE PREGLEDAČA DATOTEKA SA POVEZANOG *BLUETOOTH* UREĐAJA NA TV PRIJEMNIKU

Milan Bjelica, *Fakultet tehničkih nauka, Novi Sad*
Milan Savić, Tatjana Aleksić, *MicronasNIT, Institut za Informacione Tehnologije, Novi Sad*

Sadržaj – *U radu je predstavljena ideja, koraci razvoja i realizovano rešenje pregledača datoteka sa povezanog Bluetooth uređaja, u vidu grafičke korisničke aplikacije na TV prijemniku. Najpre je dat opšti opis sistema. Dalje, izložena je teorijska problematika, koncept rešenja i neki detalji same realizacije.*

1. UVOD

Sredinom devedesetih godina razvijeni su standardi za prenos televizijskog signala u digitalnom obliku. Digitalizacija obrade i novi pristup projektovanju TV prijemnika kao namenskog računarskog sistema, omogućila je razvoj različitih korisničkih aplikacija za TV prijemnik širokog spektra namene. Kako su moderni TV prijemnici velike dijagonale sve popularniji, pojavljuju se različite ideje za proširenja osnovne funkcionalnosti, koje koriste multimedijalne mogućnosti ovih uređaja. Osim toga, proširivanje osnovne funkcionalnosti namenskog računarskog sistema korišćenjem novih mogućnosti fizičke arhitekture i sve niže cene komponenti, predstavlja opšti trend koji nije zaobišao ni TV industriju. Uz dodatak *Bluetooth* integrisanog kola kao proširenja TV arhitekture, u ovom radu je realizovana aplikacija koja omogućava korisniku da, koristeći daljinski upravljač, posredstvom aplikacije na ekranu, obavlja *Bluetooth* povezivanje na prenosivi uređaj, pristupa sistemu datoteka tog uređaja, upravlja datotekama i pregleda ih. Tako je korisnik u mogućnosti da prikazuje slike koje je napravio mobilnim telefonom ili digitalnim fotoaparatom, ili da pregleda beleške sastavljene ručnim računarom. Korisniku su dostupne i sve mogućnosti upravljanja datotekama: kretanje kroz hijerarhiju direktorijuma, stvaranje direktorijuma, brisanje datoteka, kopiranje i premeštanje i sl. Osim toga, korisnik ima mogućnost da pokrene naizmenični prikaz slika iz odabranog direktorijuma u visokoj rezoluciji (*Slideshow*).

2. OPIS RAZVOJNE PLATFORME

Kao platforma za realizaciju rešenja, koristi se *Micronas* VGCB EVA razvojna ploča sa VGC 5969B kontrolerom, namenjenim za rad u oblasti analogne televizije, tj. za digitalnu obradu analognih video signala i njihov prikaz na ekranu. VGC 5969B (*Dual-Channel Video Graphic Controller*) je visoko kvalitetno integrisano kolo koje omogućava poboljšane dvokanalne aplikacije kao što su dvostruki prozor, slika u slici (*PIP - picture-in-picture*), slika i tekst (*PAT - picture-and-text*) čuvajući punu rezoluciju slike i poboljšavajući njen kvalitet. Velika prednost VGCB je njegov 32-bitni procesor (MIPS32TM). VGC 5969B podržava sve važne ulazne formate, a karakteriše ga 10-bitna ulazna i izlazna rezolucija i podrška prikaza do 1920x1080 tačaka. Izlazni formati uključuju 30-bitni digitalni RGB.

Kao bežični komunikacioni modul koristi se *BlueGiga WT11 Evaluation Kit wireless* komunikacioni modem. WT11 je *Bluetooth* visoko integrisano kolo koje sadrži sve potrebne elemente za *Bluetooth* komunikaciju i realizuje protokol stek zaključno sa RFCOMM nivoom, uz podršku viših nivoa u pogledu sledećih *Bluetooth* profila: DuN, HFP, OPP. Integrisano kolo radi u opsegu do 300m i poseduje 8 *MB* fleš memorije. Dvosmerni prenos podataka između TV platforme i Bluetooth modula se obavlja preko UART sprege.

Opšti prikaz sistema dat je na slici 1.

Slika 1 – *Prikaz sistema*

Polazna osnova za razvoj programske podrške je namenski operativni sistem za TV uređaje kompanije *Micronas, MICTOS* (**MIC**ronas **T**V **O**perating **S**ystem). *MICTOS* je namenjen TV uređajima zasnovanim na mikrokontrolerima kompanije *Micronas.* Pregledač datoteka dodat je kao proširenje jednostavnoj TV aplikaciji, koja je već obezbeđivala veći deo funkcionalnosti uobičajenog TV prijemnika. Komunikacija sa WT11 modulom odvija se slanjem tzv. iWRAP komandi koje podržava sam WT11. Ove komande služe za kontrolu samog integrisanog kola i izmenu njegovih parametara. Tako se iWRAP komandom CALL uz navođenje adrese *Bluetooth* uređaja u okolini može otvoriti odgovarajući RFCOMM prolaz, čime se obezbeđuju uslovi za komunikaciju sa povezanim uređajem u vidu slanja paketa odgovarajućeg nadređenog sesionog protokola.

Gore navedena fizička arhitektura i programska podrška predstavljaju dovoljnu podlogu za razvoj pregledača datoteka. S obzirom da WT11 ne podržava apstrakciju potrebnih protokola višeg OSI nivoa koji su potrebni (OBEX, OBEX FTP), isti će biti realizovani u programskoj podršci same aplikacije.

3. RAZVOJNI KORACI

Naredni koraci su bili neophodni u cilju realizacije konačnog rešenja:

- Proučavanje teorijskih osnova (protokoli OBEX i OBEX FTP, JPEG standard i dekodovanje slike);
- Realizacija i verifikacija OBEX sesionog komunikacionog protokola;
- Realizacija i verifikacija OBEX FTP aplikativnog komunikacionog protokola;
- Realizacija XMFM (*obe**X** **M**obile phone **F**ile **M**anager*) upravljača datotekama u konzoli za praćenje izvršavanja programa po uzoru na MS DOS i verifikacija svih funkcija realizovanih komunikacionih protokola;

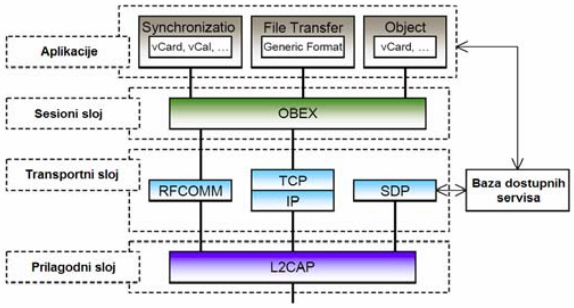
Napomene:
a) Rad je proistekao iz diplomskog-master rada Milana Bjelice. Mentor je bio prof.dr Nikola Teslić.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

- Realizacija grafičke korisničke sprege (*Graphical User Interface - GUI*) upravljača datotekama;
- Realizacija JPG pregledača slika uz integraciju JPEG dekodera;
- Integracija JPG pregledača slika unutar GUI;
- Verifikacija celokupnog programskog rešenja.

4. TEORIJSKE OSNOVE

U nastavku je predstavljen *Bluetooth* protokol stek, osnove protokola OBEX i OBEX FTP.

Bluetooth arhitektura je osmišljena u 4 nivoa po ISO OSI modelu, prikazana na slici 2.



Slika 2 – *Bluetooth protokol stek*

Gornja slika protokol steka naslonjena je na HCI (*Host Controller Interface*) koji predstavlja sam *Bluetooth* radio prenos podataka. Adaptivni sloj, L2CAP, poslednji je *Bluetooth* protokol u steku koji je specifičan za ovu tehnologiju. Transportni sloj, koji obezbeđuje pouzdan prenos podataka, realizovan je kroz RFCOMM, TCP/IP i SDP protokole, od kojih su prva dva namenjena za obezbeđivanje virtuelne serijske sprege za povezivanje uređaja uspostavljanjem virtuelne veze na zadati prolaz. OBEX, kao protokol sesionog nivoa, prvi je protokol koji definiše način komunikacije između uređaja, omogućujući im da razmenjuju podatke široke palete tipova. Konačno, aplikacioni protokoli koji se naslanjaju na OBEX omogućuju različite korisnički orjentisane upotrebe, kao što su, OBEX FTP, za razmenu i upravljanje datotekama; Sinhronizacioni protokol, za sinhronizaciju podataka između dva uređaja i sl. Logički, *Bluetooth* je hijerarhijski organizovan u profile (GAP, SPP, GOEP, FTP i dr).

OBEX protokol je jednostavan binarni komunikacioni paketski protokol za razmenu objekata podataka između dva uređaja. Oslanjajući se na transportni nivo prenosa ISO OSI modela kojim se obezbeđuje pouzdan prenos (RFCOMM u *Bluetooth* modelu), OBEX omogućava da dva povezana uređaja razmenjuju logički objedinjene celine podataka, što mogu biti datoteke, tekst ili drugi binarni podaci proizvoljnog značenja. OBEX specifikacija definiše dva glavna elementa: model reprezentacije objekata (i informacija koje opisuju objekte) i sesioni protokol koji definiše način komunikacije uređaja koji objekte razmenjuju. Model komunikacije je zahtev-odgovor (*request-response*). Tako OBEX definiše pojam klijenta kao entiteta koji otvara OBEX vezu i šalje zahteve, i pojam poslužioca koji na primljene zahteve odgovara.

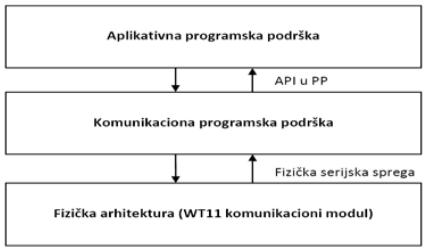
Objektni model definiše način predstave objekta unutar OBEX protokola. Model se bavi kako objektom koji se prenosi, tako i informacijama o samom objektu. Ovo se postiže tako što se svaka od informacija koja pripada objektu, uključujući i sam sadržaj objekta, smešta u posebno zaglavlje. Zaglavlje je, dakle, entitet kojim se opisuje neki od aspekata objekta, kao što su naziv, veličina, ili sam sadržaj. Sesioni protokol definiše način razmene objekata po modelu zahtev-odgovor. Tipični OBEX zahtevi su *CONNECT*, *DISCONNECT*, *PUT*, *GET*, *SETPATH* i *ABORT*. Najčešći odgovori su *OK*, za uspešno ispunjenje zahteva, *CONTINUE*, za nastavak slanja zahteva, odnosno isporuke objekta koji se sastoji iz više delova, te različiti odgovori sa kodom greške usled neuspešnog ispunjenja zahteva.

File Tranfer Profile (FTP), ili OBEX FTP, aplikativni je protokol koji se naslanja na OBEX, a koji definiše način na koji dva uređaja razmenjuju datoteke, odnosno način na koji upravljaju datotekama i pregledaju ih (*File Browsing*). Ovaj profil zadržava model klijenta i poslužioca. Po specifikaciji, FTP aplikacija mora da obezbedi odabir poslužioca sa spiska ponuđenih poslužilaca i otvaranje OBEX veze ka istom; zatim, da omogući: navigaciju direktorijuma sa prikazom hijerarhije direktorijuma koju poslužilac objavljuje, povlačenje i smeštanje objekata (datoteka ili direktorijuma), brisanje objekta, odnosno stvaranje novog direktorijuma na tekućoj putanji. Iniciranje FTP veze obavlja se uspostavljanjem OBEX zahteva *CONNECT* poslužiocu, uz naznaku usluge za navigaciju direktorijuma (*Folder Browsing Service*) u okviru zaglavlja *TARGET*: F9EC7BC4-953C-11D2-984E-525400DC9E09. Ovaj identifikator usluge je dobro poznati (*well-known*). Sam listing tekućeg direktorijuma isporučuje se u obliku objekta *x-obex/folder-listing*, koji predstavlja XML datoteku sa spiskom datoteka i direktorijuma sa informacijama. Za odgovarajuće korišćenje, dakle, potrebno je integrisati i XML parser.

5. KONCEPT REŠENJA

Kompletno rešenje podeljeno je u dve odvojene grupe:
- Komunikaciona programska podrška;
- Korisnička (aplikativna) programska podrška.

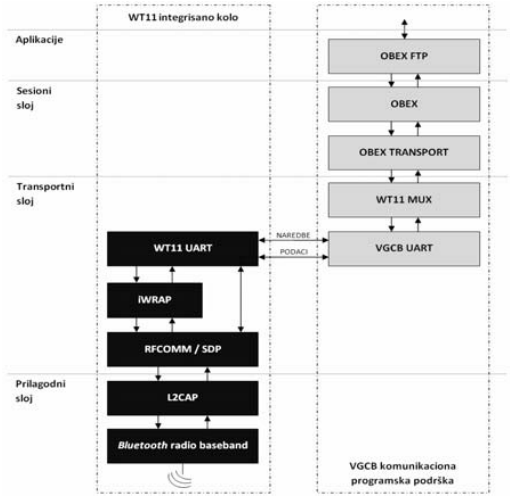
Zadatak komunikacione programske podrške je da obezbedi API (*Application Programming Interface*) sa funkcijama koje omogućuju apstrakciju mehanizama povezivanja uređaja, slanja i primanja podataka, raskidanja veze i sl. Sa stanovišta aplikacije, API treba da prikrije komunikacione detalje i da omogući jednostavan razvoj. Globalni pogled na odnos aplikativne i komunikacione programske podrške dat je na slici 3.



Slika 3 – *Organizacija programske podrške sistema*

Komunikaciona programska podrška sastoji se od podrške za komunikaciju sa komunikacionom fizičkom arhitekturom (WT11 modul). WT11 povezan je sa VGCB razvojnom pločom preko serijske UART sprege. Programska podrška u okviru MICTOS operativnog sistema obezbeđuje API za slanje i prijem podataka preko UART-a. Putem ove sprege obavlja se dvojaka komunikacija sa WT11: upravlja se radom samog integrisanog kola (slanjem iWRAP komandi), i obavlja se komunikacija sa uređajem povezanim na WT11

putem *Bluetooth* veze na RFCOMM prolazu. Ovakva dvojaka priroda komunikacije zahteva organizaciju podataka u WT11 pakete pre slanja, po definiciji proizvođača, za režim rada sa multipleksiranjem. Iz ovog razloga, uz UART komunikacioni modul programske podrške, koji je najniži u hijerarhiji, realizovan je i WT11 multipleks modul, koji organizuje podatke u WT11 pakete. Zbog potrebe za razbijanjem OBEX paketa na više delova zbog nedovoljne propusne moći WT11 integrisanog kola, uveden je i OBEX transportni modul. OBEX transportni modul i OBEX sesioni modul zajedno realizuju OBEX i čine sesioni nivo *Bluetooth* protokol steka. Oslanjanjem na ove primitive razvijen je OBEX FTP aplikativni modul. Struktura komunikacione programske podrške prikazana je na slici 4.



Slika 4 – *Komunikaciona programska podrška*

Inkapsulacija paketa kroz hijerarhiju prikazana je na slici 5.

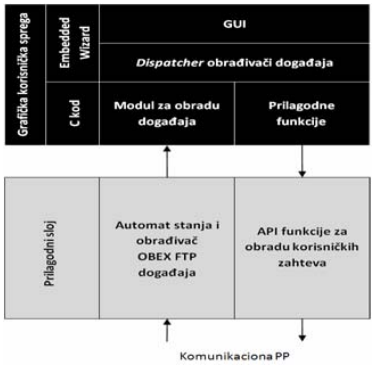
Slika 5 – *Inkapsulacija paketa*

Aplikativna programska podrška predstavlja deo realizacije okrenut korisniku. Hijerarhijska organizacija ove programske podrške osmišljena je u dve celine:

- Prilagodni sloj (funkcije koje odgovaraju na konkretne korisničke zahteve i isporučuju rezultate operacija);
- Grafička korisnička sprega (realizacija grafičkog prikaza informacija korisniku, sa mogućnošću zadavanje naredbi).

U okviru prilagodnog sloja, obezbeđeni su: odgovarajući obrađivač događaja objavljenih od strane komunikacione programske podrške (OBEX FTP modul), uz slanje poruka niti zaduženoj za grafički prikaz da isti adekvatno ažurira; slobodne funkcije koje služe za obradu korisničkih zahteva (u okviru ovih funkcija pozivaju se primitive komunikacione programske podrške); kao i konzolne komandne funkcije za omogućavanje praćenja i ispitivanja aplikacije u konzoli (već pomenuti *XMFM*).

Grafička korisnička sprega realizovana je u okruženju *EmbeddedWizard,* koje predstavlja alat za razvoj grafičke korisničke sprege. *EmbeddedWizard* koristi objektno orjentisani programski jezik *Chora* i alat za grafičko oblikovanje elemenata prikaza, čime omogućuje izradu grafičke korisničke sprege i programiranje njenog ponašanja. Ovo okruženje omogućuje automatsko generisanje koda za odredišnu platformu. U okviru EmbeddedWizard-a kreirani su dodatni oblici i klase koje podržavaju funkcionalnost upravljača datotekama, zatim, razvijen je sistem prilagodnih funkcija za komunikaciju generisanog koda sa ostatkom aplikacije, kao i obrađivač pritiska na tastere daljinskog upravljača, kako bi se nit obavestila o tim događajima. Konačno, realizovano je odgovarajuće procesiranje tih događaja u okviru *EmbeddedWizard* projekta, sa ciljem adekvatnog upravljanja GUI. Struktura aplikativne programske podrške prikazana je na slici 6.



Slika 6 – *Aplikativna programska podrška*

6. PROGRAMSKO REŠENJE

Razvoj se obavlja korišćenjem jezika C, prilagođenog razvoju za odredišnu fizičku arhitekturu TV platforme. Modularnost je postignuta pomoću sledećih funkcionalnih jedinica:

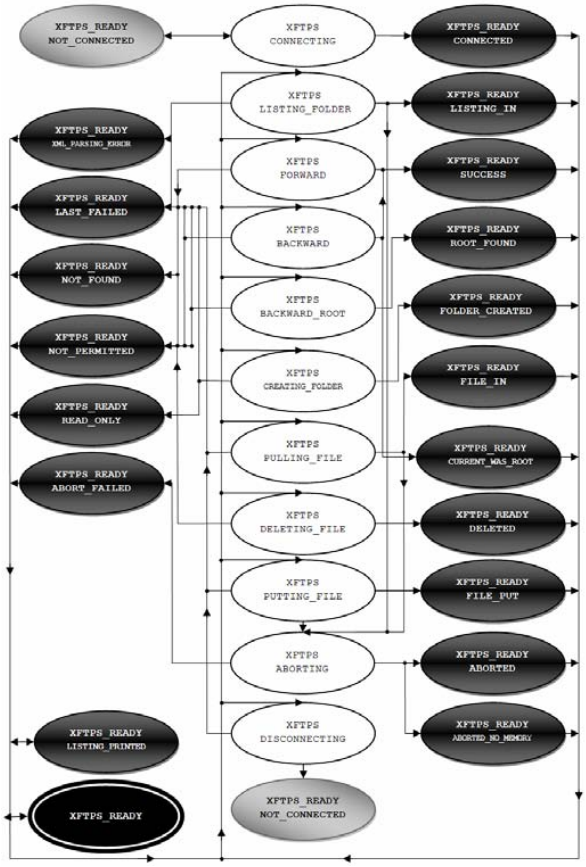
Komunikaciona programska podrška
- VGCB UART modul;
- WT11 multipleks modul;
- OBEX transportni modul;
- Modul OBEX sesionog protokola;
- Modul OBEX FTP aplikativnog protokola.

Aplikativna programska podrška
- Automat stanja i obrađivač OBEX FTP događaja, zajedno sa API funkcijama za obradu korisničkih zahteva;
- Modul za obradu događaja;
- Prilagodne funkcije za slanje zahteva sa grafičke korisničke sprege;
- EmbeddedWizard GUI i GUI obrađivači.

Dodatna programska podrška
- Rutine za XML parsiranje *folder listing* objekta;
- Rutine za dekodovanje JPG slike i njen prikaz;
- Konzolne *debug* funkcije i realizacija konzolnog upravljača datotekama.

VGCB UART modul obezbeđuje API sa funkcijama za upis podataka na UART i čitanje sa istog. WT11 multipleks modul uvodi jedan nivo apstrakcije više, organizujući podatke u WT11 pakete. Pojam OBEX paketa i rutine za slanje i prijem uvodi OBEX transportni modul. Konačno, modul OBEX sesionog protokola obezbeđuje kompletan API za uspešno kreiranje, slanje i primanje OBEX komunikacionih zahteva.

OBEX FTP aplikacioni modul obezbeđuje API za rukovanje datotekama i direktorijumima na ciljnom uređaju, realizujući FTP profil u skladu sa *Bluetooth* specifikacijom. Modul je opremljen automatom stanja (slika 7), koji je podeljen u četiri grupe stanja: blokirajuća stanja (modul je zauzet obavljanjem operacije – predstavljena belim ovalima na slici), spremno stanje bez uspostavljene veze (sivi oval), spremna stanja (modul je spreman za obavljanje sledeće operacije, dok se u kodu spremnog stanja vidi koja je operacija upravo završena – crni ovali na slici) i univerzalno spremno stanje (modul je spreman i korisnik je sigurno obavešten o ishodu poslednjeg zahteva – uokvireni crni oval). Periodičnim prozivanjem posebne funkcije u okviru ovog modula ažurira se automat stanja i obavlja sam tok komunikacije. API OBEX FTP modula obezbeđuje funkcije za iniciranje, povezivanje, postavljanje i dobavljanje datoteke, kreiranje i brisanje datoteka i/ili direktorijuma, i sl.



Slika 7 – *Automat stanja OBEX FTP modula*

Funkcionalnost komunikacione programske podrške proverena je ispitivanjem konformnosti, po OBEX specifikaciji, za slučaj realizacije klijenta, po vodiču za ispitivanje. Ispitivanje je uspešno izvršeno za sve predviđene slučajeve.

Aplikativna programska podrška ima zadatak da prati stanja odgovarajućeg automata stanja realizovanog u komunikacionoj programskoj podršci, te da u zavisnosti od istih aktivira određene akcije u cilju prikaza stanja i podataka. Dalje, ona realizuje tzv. *wrapper* funkcije oko funkcija komunikacione podrške, tamo gde je potrebno obaviti određene izmene u prikazu pri pozivu ovih funkcija. U okviru grafičke korisničke sprege realizovane u *EmbeddedWizard* okruženju kreiran je glavni oblik za upravljanje sistemom datoteka (klasa *FileBrowser*, čiji je oblik prikazan na slici 8), kao i dodatni oblici za prikaz slike i teksta, unos teksta, upit korisniku, kao i oblici za prikaz informacija o datoteci i pomoći korisniku. Konačno, kreiran je oblik za podršku naizmeničnom prikazu slika.

Slika 8 – *GUI oblik klase FileBrowser*

EmbeddedWizard projekat ne obezbeđuje sam prikaz JPG slike, već se on obavlja u prilagodnom sloju. Ovo je neophodno jer *EmbeddedWizard* okruženje omogućava prikaz slika u 8-bitnoj paleti boja, što daje neprihvatljivo loš kvalitet. Zbog toga je iskorišćen upis direktno u video memoriju TV prijemnika, u koju se inače smeštaju okviri video signala. S obzirom da je odredišna platforma dvokanalna, za prikaz slike korišćen je drugi video kanal (tzv. *slave*). Tako je omogućeno korišćenje aplikacije prikaz slika bez prekidanja prikaza video materijala koji se gleda. Da bi se slika prikazala, najpre se onemogući upis deinterlejsera u video memoriju *slave* kanala, a izlazni YUV odmerci iz JPG dekodera upisuju se direktno. S obzirom da video memorija radi po principu trostrukog baferovanja, dekodovana slika se kopira u sve bafere.

Konzolni upravljač datotekama obezbeđuje komande za upravljanje datotekama: *xconn*, *xdisconn*, *xdir*, *xdird*, *xcd*, *xcdd*, *xbd*, *xbdd*, *xroot*, *xget*, *xpaste*, *xmd*, *xdel*.

LITERATURA

[1] "irDA Object Exchange Protocol (OBEX) specification" *Infrared Data Association*, 1999.

[2] *File Transfer Profile*, Bluetooth SIG.

[3] *RFCOMM with TS 07.10,* Bluetooth SIG.

[4] *iWRAP 2.2.0 User Guide,* BlueGiga, 2007.

[5] Milan Bjelica – *Jedno rešenje Bluetooth upravljača datotekama sa pregledačem slika na TV prijemniku (interna dokumentacija),* MicronasNIT, 2007.

Abstract – This paper presents one solution of equipping a standard flat panel TV receiver with *Bluetooth* file browsing application, with a capability of previewing pictures or document files.

ONE SOLUTION OF INTEGRATED FILE BROWSER FOR CONNECTED BLUETOOTH DEVICE AS AN APPLICATION FOR TV RECEIVER

Milan Bjelica, Milan Savić, Tatjana Aleksić

JEDNO REŠENJE SMANJENJA ŠUMA VIDEO SIGNALA KORIŠĆENJEM VREMENSKO-ADAPTIVNOG FILTRIRANJA

Radmila Uzelac, Fakultet tehničkih nauka, Novi Sad,
Vladimir Zlokolica, MicronasNIT, Novi Sad
Mihajlo Katona, Fakultet tehničkih nauka, Novi Sad

Sadržaj- *Predloženi algoritam za otklanjanje šuma je adaptivan na pokret i količinu šuma prisutnog u video signalu, gde je detektor pokreta novo rešenje zasnovano na sofisticiranoj kombinaciji median i morfoloških filtara. Šum se procenjuje paralelno sa otkrivanjem pokreta, čiji rezultat se primenjuje na algoritam za odstranjivanje šuma. Šum se odstranjuje u vremenskom domenu koristeći Diskretnu Wavelet Transformaciju (DWT). Performanse algoritma za otklanjanje šuma su verifikovane u C-u i SystemC-u, gde su rezultati pokazali visoku robustnost detektora pokreta na šum i efikasno otklanjanje šuma bez unošenja artifakata.*

1.UVOD

Video sekvence imaju veliki broj aplikacija [1] počevši od televizije, preko sistema za nadgledanje, satelitskih posmatranja, tele-konferencija, video-telefona, praćenja objekata do medicinskih snimanja. Veoma bitno je da te sekvence budu što kvalitetnije, kako bi se što lakše mogli prepoznati i izdvojiti objekti ili delovi objekta posmatranja.

Tu veliki problem predstavlja šum koji se javlja kao posledica lošeg osvetljenja ili nekvalitetnih uređaja za snimanje.

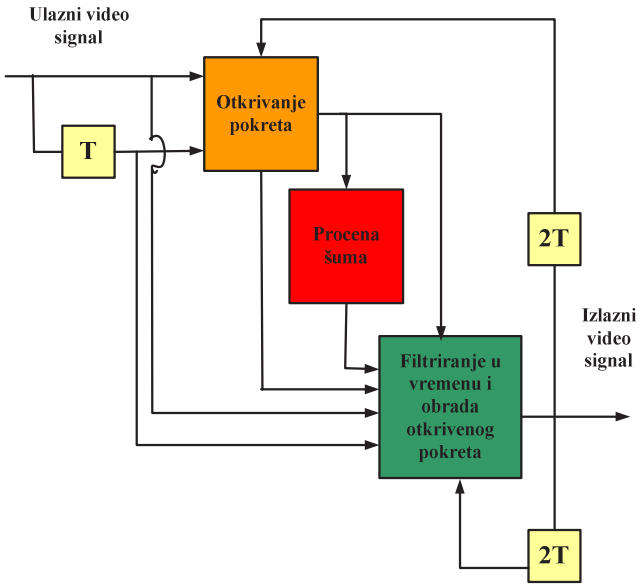
Šum se može uvesti u sekvenci za vreme akvizicije, snimanja i /ili prenosa. Izvor šuma nekada predstavlja i sama kamera (stvara tzv. akvizicioni šum), tačnije njena fizička arhitektura, a on se može još i pogoršati pod lošim uslovima osvetljenja. Šum se naročito može dodati za vreme analognog prenosa signala preko satelita ili radiodifuzijom.

Nekada je veoma bitno izvršiti smanjiti šum nad nekom video sekvencom [2], bilo zbog njene dodatne analize (kod video nadzora i praćenja objekata) ili kodovanja, bilo zbog same preglednosti i jasnoće slike. Smanjenje šuma je veoma važno kod snimanja na raznim medicinskim aparatima.

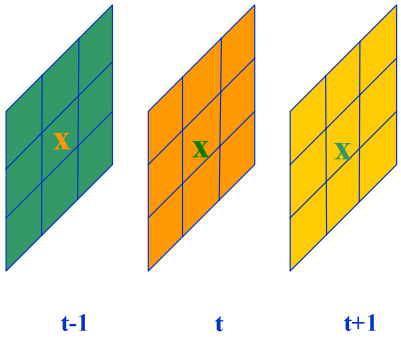
U ovom radu se daje veliki naglasak na otkrivanju pokreta, koji ima krucijalnu ulogu u otklanjanju ili smanjenju šuma, i na raznim tehnikama za otklanjanje artifakata nastalih usled otkrivanja pokreta a uzrokovanih šumom.

2. VREMENSKI-ADAPTIVNO FILTRIRANJE

Predloženi algoritam je dat na (sl.1). Kod njega se polazi od ideje da se prvo otkrije pokret, zatim proceni šum a paralelno filtrira u vremenskom domenu korišćenjem diskretne Wavelet transformacije sa jednom skalom dekompozicije, i obrađuje se samo luminansa. Proces je rekurzivan u vremenu a filtrirana slika se koristi za otkrivanje pokreta u sledećoj rekurziji.



Sl.1 *Blok dijagram generalnog algoritma*



Sl.2 *Vremenska raspodela slika*

Napomene:
a) Rad je proistekao iz diplomskog-master rada Radmile Uzelac. Mentor je bio prof.dr Vojin Šenk.
b) Rad je prethodno publikovan na konferenciji ETRAN 2008, Palić, Juni 2008.

Na osnovu [7] biće:

$$x_{L1} = \frac{1}{\sqrt{2}}(x_1 + x_0) \quad (1)$$

$$x_{L2} = \frac{1}{\sqrt{2}}(x_2 + x_1) \quad (2)$$

$$x_{H1} = \frac{1}{\sqrt{2}}(x_1 + x_0) \quad (3)$$

$$x_{H2} = \frac{1}{\sqrt{2}}(x_2 - x_1) \quad (4)$$

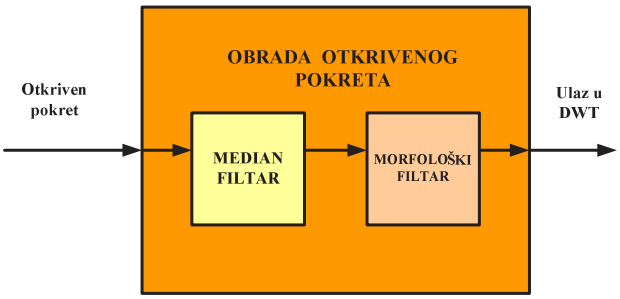
gde su x_{L1} i x_{L2} nisko-frekventni a x_{H1} i x_{H2} visoko-frekventni odbirci.

Iz priloženih formula se vidi da su potrebne tri slike da bi bila moguća obrada. Nakon iščitavanja prve tri slike u formulama (1) i (2) se mesto originalnog, zašumljenog signala sačuvanog u memoriji koristi signal dobijen primenom Haar-ovog wavelet sažimanja u vremenskom domenu na prvoj skali [7]:

$$x_R = \frac{1}{2}[\frac{1}{\sqrt{2}}(x_{L1} + x_{L2}) + \frac{1}{\sqrt{2}}(\alpha_1 x_{H1} + \alpha_2 x_{H2})] \quad (5)$$

Koeficijenti α_1 i α_2 se dobijaju na osnovu procenjenog šuma (standardne devijacije) i vrednosti obrađenog signala otkrivenog pokreta, na osnovu (10) i (11).

3. OTKRIVANJE POKRETA



Sl.3 *Blok za obradu otkrivenog pokreta*

Za detekciju pokreta koristi se apsolutna razlika vrednosti odgovarajućih piksela dveju uzastopnih slika:

$$m = | x_{trenutno} - x_{prethodno} | \quad (6)$$

m predstavlja detektovan pokret, a $x_{trenutno}$ i $x_{prethodno}$ vrednosti piksela u trenutnoj i prethodnoj slici.

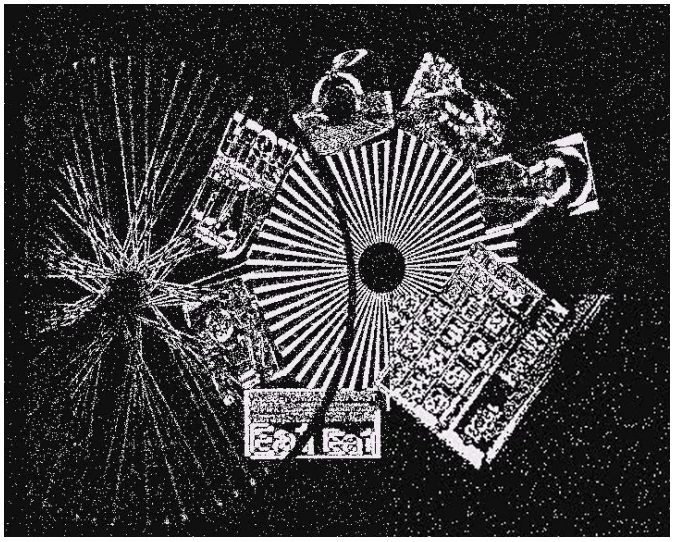
Nakon detekcije pokreta, primeni se median filter u prozoru 3x3 definisan na sledeći način:

Ako se uzme da je A={a1,a2,...,ak} skup vrednosti piksela u okolini referentnog piksela dat sa a1≤a2≤...≤ak, tada je

$$median(A) = \begin{cases} a_{\frac{k}{2}}, k = 2n \\ a_{\frac{k+1}{2}}, k = 2n+1 \end{cases} \quad (7)$$

U stvari, median skupa vrednosti je njegova centralna vrednost dobijena posle njegovog sređivanja.

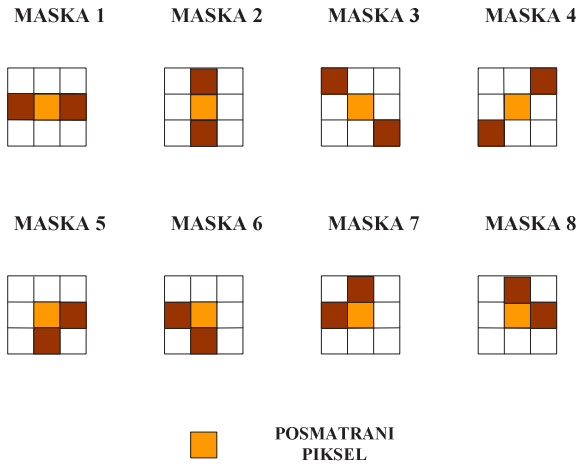
Tim otkrivene ivice objekata koji se kreću postaju kompaktnije, više liče na jednu celinu, a neke pogrešno otkrivene se odstranjuju.



Sl.4 *Primer otkrivenog pokreta, gde je belom bojom označen pokret, a crnom bojom delovi slike koji miruju*

Međutim, i posle median filtriranja mogu se primetiti otkriveni pikseli koji ne pripadaju objektu koji se krece, nego šumu (sl.4).

Zato se primenjuje postupak odstranjivanja malih regija, koji se obavlja filtriranjem pomoću morfoloških maski, [5], (sl.5):



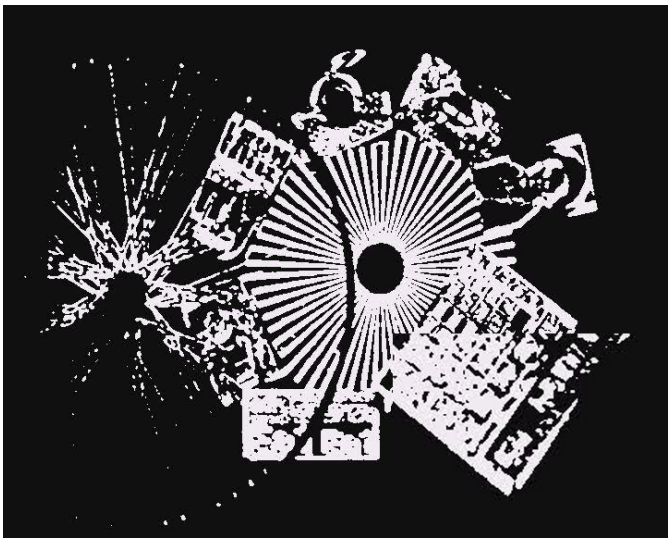
Sl.5 *Morfološke maske*

One se primenjuju prvo u morfološkoj operaciji Erozija, a zatim se rezultat podvrgne operaciji Dilacija, čime se ostvari operacija Otvaranja, koja predstavlja kombinaciju prethodne dve i dobiju rezultati približni željenim, prikazani na sl.6.
Erozija je prema [3]:

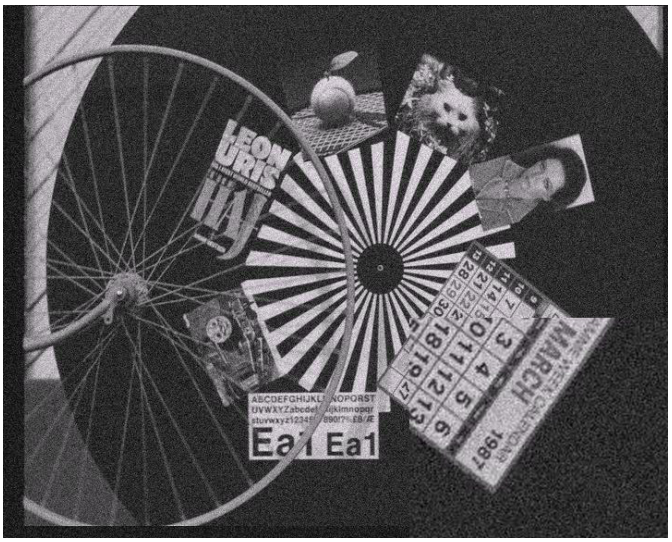
$$(f\Theta b)(s,t) = min\{ f(s+x,t+y) \quad b(x,y) \mid (s+x),(t+y) \in D_f ; (x,y) \in D_b \} \tag{8}$$

gde su D_f i D_b domeni f i b.
a Dilacija [3]:

$$(f \oplus b)(s,t) = max\{ f(s \quad x, t \quad y) + b(x,y) \mid (s \quad x),(t \quad y) \in D_f ; (x,y) \in D_b \} \tag{9}$$



Sl.6 *Izlazna slika iz morfološke operacije Otvaranje, gde je belom bojom označen pokret, a crnom bojom delovi slike koji miruju*



Sl.7 *Slika iz ulazne zašumljene sekvence*

Nakon izvršenja ovih operacija mogu se odrediti vrednosti koeficijenata α_1 i α_2 koje figurišu u (5) :

$$\alpha_1 = k_1 \frac{x_{morph1}}{\sigma} \tag{10}$$

$$\alpha_2 = k_2 \frac{x_{morph2}}{\sigma} \tag{11}$$

σ predstavlja standardnu devijaciju, koja se računa prema [4]:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2} \tag{12}$$

gde je $\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$ srednja vrednost (13)

N predstavlja onaj broj piksela u kojima postoji verovatnoća da će se dogoditi šum. Prema [6], to su pikseli koji imaju vrednost od 16 do 235, ako je maksimalna vrednost (bela boja) 255.

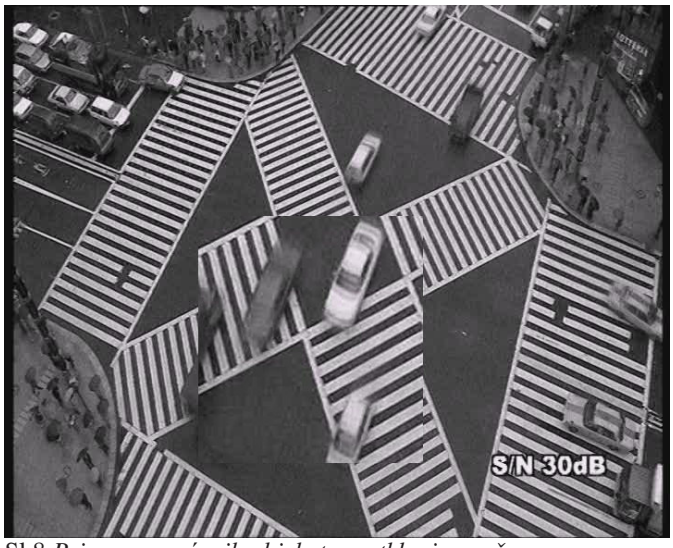
x_{morph1} predstavlja izlaz iz morfologije nakon prve dve pristigle slike, a x_{morph2} nakon druge dve slike.

Koeficijenti k_1 i k_2 se biraju tako da vrednosti dobijene pomoću (10) i (11) ne budu veće od jedan ni manje od nula.

4. REZULTATI

Algoritam je ispitan na različitim sekvencama, gde se najbolji rezultati postižu kada se algoritam izvršava nad sekvencama dobijenim snimanjem kamerom koja miruje i snima objekte koji se kreću.

Koeficijenti k_1 i k_2 treba da se izaberu tako da bi se dobila izvesna ravnoteža izmedju količine uklonjenog šuma i zamućenja objekata koji se kreću (sl.8).



Sl.8 *Primer zamućenih objekata a otklonjenog šuma*

Sl.9 *Ulazna sekvenca zašumljena ne uniformnim šumom*

Sl.10 *Izlazna slika dobijena pažljivim odabirom koeficijenata*

5. ZAKLJUČAK

U ovom radu je predstavljeno jedno rešenje za smanjenje šuma video signala korišćenjem vremensko-adaptivnog filtriranja.

Rad je verifikovan u programskim jezicima C i SystemC, gde je pokazano da zadovoljavajuće smanjuje šum zastupljen u obrađivanim video sekvencama.

Pošto se najbolje performanse dobijaju nad video sekvencama dobijenih snimanjem nepokretnom kamerom, jedna od primena bi mogla biti u sistemima za nadzor i praćenje objekata.

Realizacija algoritma je prilagođena implementaciji u FPGA ili CELL platformi, čime se može ispitati njegovo funkcionisanje u okruženju u realnom vremenu.

LITERATURA

[1] Vladimir Zlokolica , «Advanced Nonlinear Methods for Video Denoising « , Phd. Thesis , 2006.

[2] Francesco Cocchia, Sergio Carrato, and Giovani Ramponi ,«Design and Real-Time Implementation of 3-D Rational Filter for Edge Preserving Smoothing “, IEEE Transactions on Consumer Electronics , Vol. 43, No.4, November 1997.

[3] Raphael C.Gonzlez ,Richard E.Woods, «Digital Image Processing «, Second Edition, 2002.

[4] Mila Stojaković, «Slučajni Procesi «, Univerzitet u Novom Sadu, 2000.

[5] Aishy Amer,”Fast and Reliable Structure-Oriented Video Noise estimation”, IEEE Transactions on circuits and systems foe video technology, vol.15(1), 2005.

[6] Robert Thoma, Mathias Bierling, “Motion Compensating Interpolation Considering Covered and Uncovered Background”, 1989.

[7] D.Donoho, «De-Noising by soft thresholding», IEEE Transactions on Information Theory, vol.38(2), pp.613-627, 1995.

Abstract- *This paper presents a new algoritm for motion and noise adaptive temporal denoising. The proposed motion-detection scheme is a novel algorithm based on a sofisticated combination of median and morfological filters. Noise estimation is performed in parallel to the motion detection and results are applied to the denoising algorithm. Denoising is performed in time domain using Discrete Wavelet Transformation(DWT). Denoising performance is verified in C and SystemC against robustness to noise and motion blur.*

ONE SOLUTION FOR DE-NOISING OF VIDEO SIGNALS USING TIME-ADAPTIVE FILTERING

Radmila Uzelac, Vladimir Zlokolica,
Mihajlo Katona

AUTOMATSKO GRUPISANJE REČI PO SEMANTIČKOJ SLIČNOSTI POMOĆU K-MEANS ALGORITMA

Atila Farkaš, *Fakultet tehničkih nauka u Novom Sadu*

***Sadržaj* — Za razumevanje prirodnog jezika od strane nekog automatskog sistema potrebno je predstavljanje značenja na taj način da ona bude upotrebljiva od strane automatskog sistema. Automatsko pronalaženje mere sličnosti neke nove reči u odnosu na neku poznatu reč je mnogo jednostavnije nego određivanje njenog tačnog značenja. U radu je opisana primena k-Means algoritma za automatsko grupisanje reči po semantičkoj sličnosti. Argumenti k-Means algoritma su vektori koji predstavljaju semantičku povezanosti reči.**

***Ključne reči* — Obrada prirodnog jezika, automatsko grupisanje po semantičkoj sličnosti, k-Means algoritam.**

I. UVOD

OBRADA prirodnog jezika je oblast veštačke inteligencije i lingvistike, koja se bavi proučavanjem automatskog generisanja i razumevanja prirodnog ljudskog jezika. Sistemi za generisanje prirodnog jezika pretvaraju informacije iz računarske baze podataka u ljudski jezik koji prirodno zvuči a sistemi za razumevanje prirodnog jezika pretvaraju primere ljudskog jezika u više formalne predstave sa kojima računarski programi lakše manipulišu. Obrada prirodnog jezika je veoma privlačan metod interakcije između čoveka i računara. Programi koji se bave obradom prirodnog jezika mogu da služe i samo kao interfejs nekom primarnom programu [1]. Npr. prirodno jezički interfejs za sistem baza podataka prevodi ulaz od strane korisnika u formalni upit u baze i zatim sistem nastavlja obradu bez dalje potrebe za tehnikama obrade prirodnog jezika.

Veštačka inteligencija kao pojam u širem smislu, označava kapacitet jedne veštačke tvorevine za realizovanje funkcija koje su karakteristika ljudskog razmišljanja, kao npr. razumevanje prirodnih (i veštačkih) jezika. U procesiranju informacija koncentriše se na programe koji nastoje da osposobe računar za razumevanje pisane i verbalne informacije, stvaranje rezimea, davanje odgovara na određena pitanja ili redistribuciju podataka korisnicima zainteresovanim za određene delove tih informacija. U tim programima, od suštinskog je značaja, kapacitet sistema za stvaranjem gramatički korektnih rečenica i uspostavljanje veze između reči i ideja, odnosno identifikacija značenja. Dok je problem strukturne logike jezika, odnosno njegove sintakse, moguće rešiti programiranjem odgovarajućih algoritama, problem značenja, ili semantike, je mnogo dublji i ide u pravcu autentične veštačke inteligencije.

Sistem mora da komunicira sa čovekom i drugim inteligentnim sistemima na "prijateljski način" - zato treba da upotrebljava prirodni jezik i govor. Takva komunikacija podrazumeva baratanje i dvosmislenostima i gramatički neispravnim rečenicama, tolerisanje grešaka i nejasnoća u komunikaciji [2]. Praktični sistemi za obradu prirodnih jezika moraju donositi ne dvosmislene odluke kod značenja reči, kategorija reči, kod sintaksne strukture i u semantičkom domenu.

II. KLASTEROVANJE (GRUPISANJE)

Klasterovanje predstavlja particionisanje skupa objekata u podskupove (grupe – klastere), tako da objekti u podskupovima dele neko zajedničko obeležje. Cilj algoritma za klasterovanje je da svrsta objekte sa sličnim osobinama u istu grupu.

Jedan od najjednostavnijih algoritama je k-Means algoritam koji vrši nehijerarhijsko hard klasterovanje objekta, što znači da jedan objekat može pripadati samo jednom klasteru.

Klasteri su definisani centrima koje određuju njihovi članovi. Prvo se proizvoljno izaberu inicijalni centri klastera. Zatim se prolazi kroz nekoliko iteracija tokom kojih se objekti pridružuju klasterima čijim centrima su najbliži. U svakoj iteraciji, kada su svi objekti pridruženi, preračunava se centar svakog klastera kao srednja vrednost njegovih članova:

$$\vec{\mu} = \frac{1}{|c_j|} \sum_{\vec{x} \in c_j} \vec{x} \qquad (1)$$

Članovi (objekti) su predstavljeni pomoću vektora $\vec{x} = (x_1, x_2, ..., x_n)$

a $|c_j|$ je broj elemenata j-tog klastera, c_j.

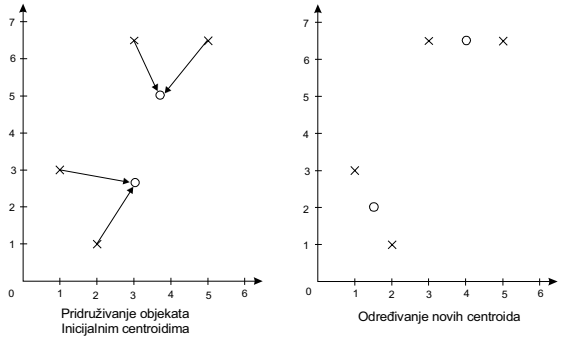
Dobijeni centri su poznati kao centroidi i oni se najčešće ne poklapaju sa objektima unutar te klase

Sl. 1. prikazuje primer jedne iteracije k-Means algoritma. Prvo se objekti pridruže klasterima čijim centrima su najbliži. Zatim ponovo određuju centri kao srednja vrednost njihovih članova. U ovom slučaju, dalja iteracija ne bi promenila klastere, s obzirom da

Atila Farkaš, Veljka Vlahovića 18, 21220 Bečej, Srbija; (telefon: +381-63-863-77-36; e-mail: atilaf@yahoo.com).

Napomene:
a) Rad je proistekao iz diplomskog-master rada Atile Farkaša. Mentor je bio prof.dr Vlado Delić.
b) Rad je prethodno publikovan na konferenciji TELFOR, Beograd, Novembar 2008.

pridruživanje novim centrima ne menja pripadnost klasteru ni jednog objekta. To znači da se ni centri ne bi promenili u sledećem preračunavanju. Ali ovo nije opšti slučaj. Najčešće je potrebno nekoliko iteracija da bi algoritam konvergirao [3].



Sl. 1. Jedna iteracija k-Means algoritma

Kao rastojanje može se koristiti i kosinus ugla između vektora kojima su opisani objekti i vektora koji predstavlja centroid:

$$\cos(\vec{x},\vec{y}) = \frac{\vec{x}\cdot\vec{y}}{|\vec{x}|\cdot|\vec{y}|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \qquad (2)$$

Određivanje klastera sa što boljim rasporedom objekata moguće je minimiziranjem nekih od sledećih metrika:

- maksimalno rastojanje objekata od svojih centroida
- zbir srednjih vrednosti rastojanja od centrioda za sve kalstere
- zbir varijansi po svim klasterima
- ukupno rastojanje objekata od svojih centroida

Često se ne zna unapred tačan broj klastera. Gore navedene metrike mogu posluže i kod određivanja broja klastera n, tako što će se algoritam izvršavati više puta za različito n dok mera za kvalitet ne bude zadovoljavajuća.

Kod k-Means algoritma inicijalni centri se biraju proizvoljno. U zavisnosti od strukture skupa objekata ovaj izbor može biti važan. Male promene kod inicijalnih centroida mogu dati skroz različite rezultate klasterovanja [4]. Izvršavanje algoritma se može ponoviti više puta za različit izbor početnih centorida i da se na kraju izabere najbolji rezultat, za koji je npr. minimalna neka od gore navedenih metrika. Na rezultat takođe može da utiče i redosled objekata koje grupišu.

A. Primena k-Means algoritma za grupisanje reči

k-Means algoritam za klasterovanje može da se primeni i na rečima prirodnog jezika. Reči, čije značenje je na neki način povezano, svrstavaju se u istu grupu. Kriterijum "sličnosti" može biti semantička sličnost. Pod semantičkoj sličnosti se ne misli samo na sinonime, već se smatra da su neke reči slične ako pripadaju istoj semantičkoj kategoriji (sadržini). Npr. lekar, ambulanta, operacija, mogu pripadati jednoj grupi semantički sličnih reči.

Smatraće se da su reči slične ukoliko postoji verovatnoća da se one pojave zajedno. Pored velikog broja načina određivanja mere semantičke sličnosti, najbolje može da se shvati pomoću sličnosti vektora. Reči, čija semantička sličnost želi da se odredi, predstavljaju se pomoću vektora u višedimenzionalnom prostoru.

Algoritam treba da bude potpuno automatski i da se oslanja na statističku analizu vrlo velikog broja tekstova. Nakon analize, formira se matrica na osnovu broja dokumenta u kojima se pojavljuju obe reči Vektor vrste ove matrice predstavljaju reči kao vektore. Primer matrice prikazan je u Tabeli 1. Element a_{ij} predstavlja broj dokumenta u kojima su se i *i*-ta i *j*-ta reč pojavili u isto vreme. Reči se smatraju sličnim ako se one pojavljuju zajedno u dokumentima. Međutim, reči mogu biti slične i kada se one ne pojavljuju zajedno, ali se javljaju u društvu neke treće reči, koja na taj način čini vezu između njih.

TABELA 1: MATRICA ZAJEDNIČKOG POJAVLJIVANJA REČI

i \ j	parlament	ministar	vlast	lopta	utakmica	turnir
parlament	30	18	17	0	0	0
ministar	18	47	15	0	1	0
vlast	17	15	33	0	0	0
lopta	0	0	0	21	16	4
utakmica	0	1	0	16	64	21
turnir	0	0	0	4	21	36

Zajednička pojavljivanja mogu da se definišu nad dokumentima, paragrafima ili nad nekim drugim jedinicama. Različit način prikaza reči u vektorskom prostoru, daje različite semantičke sličnosti.

U navedenom primeru uzete su samo 6 reči koje se mogu podeliti u dve grupe - sport ili politika. Algoritam, međutim ne treba da zna koje kategorije predstavljaju ove grupe. Algoritmu takođe nije poznat ni broj grupa, već se ona zadaje ili određuje tako da se dobije najbolji rezultat koji zadovoljava neki kriterijum ili minimizira neku od pomenutih metrika. Najbolji izbor za broj klastera (grupa) može da se odredi i ekperimentisanjem. Pri analizi rezultata klasterovanja za različiti broj klastera, dolazi se do zaključka da se optimalnim izborom broja klasetra smanjuje uticaj početnog izbora centroida kao i redosleda reči koji se grupišu, na rezultat klastrovanja k-Means algoritmom.

Tabela 1. dobijena je analizom 250 dokumenata, u proseku sa oko 600 reči po dokumentu. Vektori koji predstavljaju reči *parlament, ministar, vlast, lopta, utakmica, turnir* su redom:

$$\vec{x}_1 = (30,18,17,0,0,0) \qquad \vec{x}_4 = (0,0,0,21,16,4)$$
$$\vec{x}_2 = (18,47,15,0,1,0) \qquad \vec{x}_5 = (0,1,0,16,64,21)$$
$$\vec{x}_3 = (17,15,33,0,0,0) \qquad \vec{x}_6 = (0,0,0,4,21,36)$$

Izbor inicijalnih centorida je proizvoljan i oni mogu da se poklapaju npr. sa vektorima $\vec{x}_1$ i $\vec{x}_2$:

$$\vec{\mu}_1 = (30,18,17,0,0,0)$$
$$\vec{\mu}_2 = (18,47,15,0,1,0)$$

Nakon prve iteracije, vektori $\vec{x}_1$ i $\vec{x}_3$ će pripasti prvom klasteru c_1, pošto će ugao između ovih vektora i prvog centroida $\vec{\mu}_1$ biti manji od ugla koji zaklapaju su sa drugim centroidom (fromula (2)). Analogno će vektori $\vec{x}_2$, $\vec{x}_4$, $\vec{x}_5$, $\vec{x}_6$ pripasti drugom klasteru c_2.

Pomoću formule (1) odrede se novi centroidi:

$$\vec{\mu}_1 = (23,16,25,0,0,0)$$
$$\vec{\mu}_2 = (4,12,3,10,25,15)$$

Nakon druge iteracije raspored postaje:

$$c_1 = \{\vec{x}_1, \vec{x}_2, \vec{x}_3\} \text{ i } c_2 = \{\vec{x}_4, \vec{x}_5, \vec{x}_6\}$$

a novi centroidi su:

$$\vec{\mu}_1 = (21,26,21,0,0,0)$$
$$\vec{\mu}_2 = (0,0,0,13,33,20)$$

U svakoj narednoj iteraciji centroidi ostaju isti, pošto vektori (reči) ne menjaju dalje pripadnost klasterima i time se i završava iteracija. Rezultat klasterovanja u dve grupe je dakle:

1. grupa: parlament, ministar, vlast
2. grupa: lopta, utakmica, turnir.

III. ZAKLJUČAK

Određivanjem pripadnosti reči klasterima, dobijaju se bliže informacije o značenju neke reči. Većina osobina reči koja su interesantna za obradu prirodnog jezika, nisu u potpunosti pokrivena u rečnicima koje mašine mogu da čitaju. Razlog je produktivnost prirodnog jezika. Stalno se pronalaze nove reči ili se stare reči koriste sa novim značenjem. Čak i kada bi se napravio rečnik koji bi potpuno pokrivao jezik današnjice, za nekoliko meseci taj rečnik bi neizbežno postao nekompletan. Grupisanjem se pomaže pronalaženje osobina i značenja reči, što je značajno u inteligentnim računarskim sistemima, gde je lakše računaru opisati povezanost sa drugim rečima nego značenje same reči.

LITERATURA

[1] Thomas C. Rindflesch: "Natural Language Processing", Sematic Knowledge Representation research paper, USA, 1996.

[2] L. Budin, B.Dalbelo Bašić, S. Ribarić, N. Pavešić: "Inteligentni sustavi", međunarodna konferencija MIPRO Opatija, 2001.

[3] Christopher D. Manning, Hinrich Schütze: „Foundation of Statistical Natural Language Processing", Massachusetts Institute of Technology, London, Second printing with correction 2000.

[4] Ian H.Witten and Eibe Frank: „Data Mining: Practical Machine Learning Tools and Techniques", Morgan Kaufmann Publishers, Second Edition; 2005.

[5] Dan W. Patterson: „Introduction to Artificial Intelligence and Expert Systems", Prentice Hall, 1990.

ABSTRACT

For understanding of natural language by automatic system, the meaning has to be represented in way which can be used by the system. Finding measure of similarity of words is more easier then determine the exact meaning. This paper describe implementation of k-Means algorithm for grouping words based on semantic similarity.

AUTOMATIC CLUSTERING OF SEMANTICALLY SIMILAR WORDS USING k-MEANS ALGORITHM

Atila Farkaš.

UTICAJ REMONTA DALEKOVODA NA PRORAČUN PREKOGRANIČNIH PRENOSNIH KAPACITETA

INFLUENCE OF POWER LINES MAINTAINANCE ON CROS-BORDER TRANSMISSION CAPACITY

Andrija Oros, *Fakultet tehničkih nauka, Novi Sad*

Oblast – *ELEKTROTEHNIKA I RAČUNARSTVO*

Kratak sadržaj – *Sa trendom liberalizacije energetskog tržišta i povećanjem obima razmene trgovine električne energije između sistema, postavljaju se novi ciljevi poslovanja u domenu elektroprivrede. Kamen spoticanja trenda ekonomskog upliva u sferu energetike tehničke je prirode, definisan kroz pojmove* prenosni kapacitet *i* zagušenja, *koji se javljaju u prenosnoj mreži. Značajni uticaj na ova dva ograničavajuća faktora ima* remont *prenosnih elemenata sistema. Primenom savremenih softverskih alata za modelovanje sistema i NTC metodologijom utvrđuju se mogućnosti sistema za prenos i razmenu energije, pod promenljivim uslovima usled remonta i različitih neraspoloživosti, uz održavanje kriterijuma sigurnosti i pouzdanost sistema.*

Abstract – *In light of electricity market liberalization and increase of scheduled exchange between systems, new goals are set in electric power business. Main barrier for economic impact in field of electric power is matter of technical issue, defined within concept of transfer capacity and congestion in transmission lines. Maintenance of transmission elements of the power system has significant influence within these two new concepts. Modern software tools for system modeling and NTC methodology opened possibilities for evaluation of energy transport and exchange, under variable conditions of maintenance and possible contingencies, within system reliability and security standards.*

Ključne reči: *Remont, dalekovod, prenosni kapacitet.*

1. UVOD

Jedan od sastavnih delova elektroenergetskog sistema (EES) koji umnogome određuje njegov značaj i kvalitet, jeste prenosna mreža, koja povezuje elektrane i potrošačke čvorove. U prošlosti osnovni cilj svakog EES bio je da sistem bude što je moguće duže *samoizbalansiran*, a interkonektivni dalekovodi su prvenstveno služili za povezivanje (nacionalnih) EES sa ciljem da se obezbedi veća sigurnost, pouzdanost i bolja regulacija. Na tržištu su se pojavili novi učesnici, kako na nacionalnim, tako i na regionalnom, odnosno evropskom tržištu električne energije. Posledica je značajno povećanje broja transakcija električne energije, koje je teško u svakom trenutku fizički koordinisati i tehnički realizovati kako bi se obezbedila zahtevana sigurnost prenosne mreže.

Cilj rada je analiza uticaja remonta dalekovoda na proračun prenosnih prekograničnih kapaciteta. Opisu standardnih procedura održavanja pridodaju se oblici neraspoloživosti *n-1-1 kriterijum*, tj. *održavanje/ispad neraspoloživost* prenosnih elemenata kao osnovni faktor za analizu u proračunima i ispitivanje njihovih uticaja na sistem. Uticaj remonta dalekovoda na proračun prekograničnih prenosnih kapaciteta dat je kroz sintezu tehničkih kriterijuma i ekonomskih metoda kroz proračun savremenim softverskim alatima.

2. REMONT DALEKOVODA I KRITERIJUMI SIGURNOSTI

Savremeni tehnički sistemi zahtevaju intenzivno i neprekidno održavanje, kako bi zastoji u proizvodnom procesu bili izbegnuti. Proces remonta elemenata EES mora ispoštovati tehničke procedure remonta, sigurnost sistema i neprekidnost napajanja potrošača, kao i neometano funkcionisanje tržišta električne energije. Najvažniji zadatak planiranja održavanja jeste donošenje odluke o periodu u kome treba da se sprovede održavanje. Planirani remont prenosnih elemenata sistema odvija se u letnjem periodu, u mesecima od marta do novembra, kada je smanjena potrošnje električne energije, kako bi uticaj na funkcionisanje sistema bio minimalan. Pravilan izbor intervala remonta elemenata prenosnih i proizvodnih kapaciteta potrebno je usaglasiti kako unutar jednog TSO-a (Transmission System Operator – Operator Prenosnog Sistema), tako i u okvirima regiona, tako da dispečerski centri u regionu zajednički u koordiniranom dogovoru vrše planiranje rasporeda remonta značajnih prenosnih i proizvodnih elemenata (elementi naponskog nivoa 220kV i 400kV).

Pouzdan rad EES osnovni je zadatak TSO-a. On se ostvaruje zadovoljenjem sledećih tehničkih kriterijuma: *n-1 kriterijum*, naponski i kriterijum reaktivne snage, kriterijum kratkih spojeva i stabilnosti sistema. Kriterijum n-1 u prenosnoj mreži je ispunjen ukoliko su, usled jednostrukog ispada jednog elementa – nadzemnog voda, kabla, mrežnog transformatora ili generatora priključenog na prenosnu mrežu – zadovoljeni svi sigurnosni kriterijumi. *Kriterijum n-1-1 ispad/održavanje* podazumeva proveru *n*-1 kriterijuma (ispad bilo kog elementa) kada je jedan element u stanju neraspoloživosti usled remonta.

3. ALOKACIJA PRENOSNIH KAPACITETA I UPRAVLJANJE ZAGUŠENJIMA

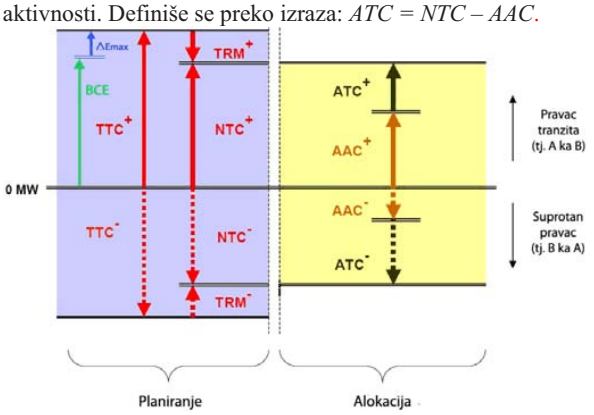
Udruženje evropskih operatora prenosnih sistema (ETSO) odredilo je definicije prenosnih kapaciteta koji se koriste za proračune internacionalne razmene električne energije između članova ETSO (slika 1):

Ukupni prenosni kapacitet – TTC predstavlja maksimalni mogući program razmene snage između dve regulacione oblasti, koji može da se odvija kontinualno, bez narušavanja postavljenih standarda pogonske sigurnosti sistema, primenljiv u svim sistemima interkonekcje, ako je buduće stanje mreže unapred poznato. *Margina pouzdanosti prenosa – TRM* predstavlja sigurnosnu marginu prenosa snage, usled

Napomena:
a) Rad je proistekao iz diplomskog-master rada Andrije Orosa. Mentor je bio prof.dr Ljubomir Gerić.

nepreciznosti u proračunu *TTC* vrednosti, devijacije stvarnih tokova snaga, poremećaja balansa proizvodnje i potrošnje. TRM obezbeđuje neophodnu rezervu u prenosnom kapacitetu za rad primarne, sekundarne i tercijarne regulacije. *Neto prenosni kapacitet* (*NTC*) predstavlja maksimalni mogući program razmene snage između dve regulacione oblasti, kompatibilan sa standardima pogonske sigurnosti, primenljiv u svim sistemima interkonekcije, uzimajući u obzir sigurnosnu marginu i tehničke nesigurnosti: $NTC = TTC - TRM$.

Iskorišćeni deo prenosnog kapaciteta (*AAC*) predstavlja ukupnu količinu već zauzetih prenosnih prava. *Raspoloživi prenosni kapacitet* (*ATC*) je neangažovani deo *NTC*-a, koji stoji na raspolaganju u toku trenutne faze utvrđene procedure alokacije prenosnih kapaciteta, u svrhu komercijalnih aktivnosti. Definiše se preko izraza: $ATC = NTC - AAC$.



Slika 1: Definicije prenosnih kapaciteta

4. METODOLOGIJA PRORAČUNA

Proračun TTC, za dve susedne oblasti A i B, započinje postavljanjem osnovnog plana razmene (*BCE*). Osnovni plan razmene sadrži plan proizvodnje i potrošnje, kao i različite transakcije (od dugoročnih do *spot* ugovora). Promenom proizvodnje određuje se *NTC*-a na sledeći način: U oblasti jednog TSO-a aktivna snaga generatora se povećava, dok se u oblasti drugog aktivna snaga generatora smanjuje za istu vrednost ΔE istovremeno. ΔE se povećava iterativno dok se n-1 kriterijum ne naruši, bilo u sistemu A ili B. Kao rezultat se dobijaju vrednosti $\Delta Emax^{+}$. Maksimum razmene iz oblasti A u B, saglasan sa sigurnosnim kriterijumima bez uzimanja u obzir neizvesnosti i nepreciznosti, odnosno ukupani prenosni kapacitet (*TTC*) od A do B tada iznosi $BCE + \Delta Emax^{+}$.

Procedura je ista u suprotnom smeru – spuštanje proizvodnje u sistemu A i povećanje proizvodnje u sistemu B – prilikom proračuna *TTC* iz oblasti B prema oblasti A

Razlika uvezene i izvezene električne energije iz sistema predstavlja bilans, dok se *tranzit* električne energije kroz EES izračunava kao minimum apsolutne vrednosti pojedinačne sume izvoza i sume uvoza električne energije u EES:

$$\text{Tranzit} = \min\left[\left|\sum_{t=1}^{8760}\text{Uvoza}\right|,\left|\sum_{t=1}^{8760}\text{Izvoza}\right|\right] \quad (1)$$

Svojim geografskim položajem Srbija se u regionu ističe kao najznačajnija zemlja za tranzit električne energije sa dva glavna tranzitna pravca: Prvi je od severa prema jugu, tj. iz Mađarske, Bugarske i Rumunije prema jugoistočnoj Evropi i drugi dominantni pravac od istoka prema zapadu, gde Bugarska i Rumunija izvoze znatnu količinu energije prema Hrvatskoj, Crnoj Gori, Albaniji i Makedoniji.

Interkonektivni EES nose sa sobom određene specifičnosti koje je potrebno uzeti u obzir kako u upravljanju tako i kod samog proračuna prenosnih kapaciteta. U okviru tokova snaga dva susedna sistema u osnovnom scenariju uvek postoji i fenomen paralelnog (kružnog) toka snaga preko „trećih“ sistema. Posledica navedene osobine sistema u interkonekciji jeste razlika u vrednostima programskih veličina i stvarnih fizičkih vrednosti u sistemu.

Zagušenjem u prenosnoj mreži naziva se stanje prenosnog sistema u kojem proizvođači ili potrošači električne energije žele proizvesti i potrošiti električnu energiju na način koji bi uzrokovao pogon prenosnog sistema na granici jednog ili više ograničenja. Do zagušenja može doći kako na poveznim dalekovodima između susednih sistema tako i unutar samih sistema. Metode kojima se upravlja zagušenjima moraju biti nepristrasne, ekonomski efikasne, transparentne i primenljive sa različitim vrstama ugovora i transakcija koje postoje na različitim tržištima. *Eksplicitna aukcija* predstavlja metod koji se koristi za upravljanje zagušenjima u EES na mesečnom nivou. U okviru ovog mehanizma, odvojeno se trguje kapacitetom i električnom energijom (metoda je „eksplicitna“). *TSO*-i proračunavaju i objavljuju *NTC*, a učesnik na tržištu nudi svoju cenu za korišćenje *NTC*-a. U narednom koraku, ponude svih učesnika se sakupljaju i rangiraju respektujući kriterijum ponude sa većom cenom. Ponude se redom prihvataju, sve dotle dok se *NTC* kompletno ne iskoristi. Dva su osnovna načina za određivanje cene poravnanja po kojoj će se naplaćivati prenosni kapacitet, odnosno zakup za njegovo korišćenje: Svako plaća onoliko koliko je ponudio za pravo prenosa (*pay as bid*), ili se određuje marginalna cena tako što cena ponude koja je “poslednja” ušla u trku, odnosno najniža cena među prihvaćenim transakcijama, koju plaćaju svi učesnici čije su ponude prihvaćene (*marginal pricing*). Kod ove metode važi princip “iskoristi ili izgubi” (*use it or lose it*), odnosno alocirani kapacitet iz prethodnih aukcija (na dužem vremenskom horizontu) čije korišćenje nije potvrđeno, dodeljuje se drugim učesnicima.

5. PRORAČUN PRENOSNIH KAPACITETA PRI n-1-1 KRITERIJUMU

Programski paket Power System Analyzer razvijen je od strane Elektroenergetskog Koordinacionog Centra (EKC, Beograd), za proračun svih statičkih analiza tokova snaga i provere sigurnosnih kriterijuma u radu EES u skladu sa zahtevima procedura definisanih od strane UCTE-a (Union for the Co-ordination of Transmission of Electricity) i ETSO-a. PSA je u osnovi program za proračune tokova snaga, baziran na punoj Newton-Raphson metodologiji. Pored osnovnih programskih alata za unošenje i editovanje elementa modela, u programu su implementirani i alati za automatsku proveru kriterijuma n-1, proračune *NTC*-a, linearno podizanje i spuštanje proizvodnje i potrošnje celog modela ili pojedinih oblasti. Izvršeno je modelovanje čitave prenosne mreže jugoistočne Evrope definisanjem veličina i karakteristika svakog elementa ponaosob (čvorova, grana i transformatora). Njihovim povezivanjem formira se mreža koja mora verno prikazati stanje u sistemu. Postavku čini *osnovni plan rada (base case)* sistema za leto, stanje za karakteristični dan u nedelji (treća sreda za izabrani mesec). Postupak proračuna u PSA se svodi na ispitivanje n-1-1 kriterijuma, odnosno dalekovodi se postavljaju u stanje remonta i vrši se proračun prenosnih kapaciteta povećanjem/spuštanjem proizvodnje u dva susedna sistema dok se ne naruši n-1 kriterijum u posmatranom sistemu. Za procenu uticaja remonta dalekovoda na vrednosti prenosnih kapaciteta (*TTC*-a), izabrana je grupa dalekovoda prikazana slikom, pet unutrašnjih i interkonektivnih dalekovoda naponskog nivoa 400 kV (tabela 1).

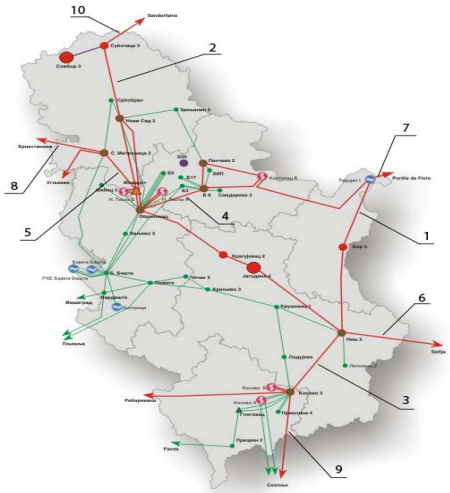
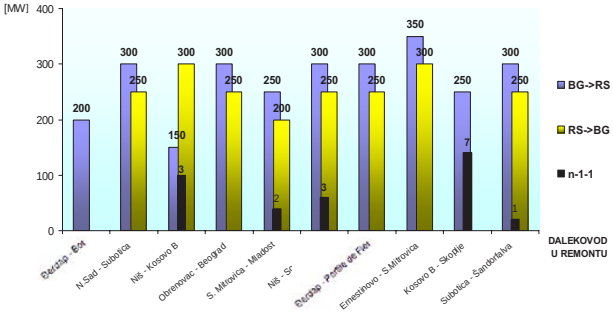


Tabela 1: Izabrani dalekovodi 400kV

Unutrašnji dalekovodi:	Interkonektivni dalekovodi
1. Đerdap – Bor	6. Niš – Sofija West
2. Novi Sad – Subotica	7. Đerdap – Portile de Fier
3. Niš - Kosovo B	8. Ernestinovo – S.Mitrovica
4. Obrenovac – Beograd	9. Kosovo B – Skoplje
5. S. Mitrovica – Mladost	10. Subotica – Šandorfalva

Proračun se obavlja između prenosnog sistema Srbije i svakog susednog sistema pojedinačno – ukupno osam podsistema interkonekcije sa dva smera proračuna. Za oba sistema koristi se metod promene proizvodnje (ΔE) – proporcionalno rezervi, generatori raspodeljuju zadatu promenu proizvodnje proporcionalno sopstvenoj raspoloživoj rezervi. Kada se n-1 kriterijum naruši, beleži se poslednja vrednost ΔE, koja postaje $\Delta Emax$, odnosno dobija se maksimalni iznos energije koja može da se transportuje bez narušenja sigurnosnih kriterijuma u susedni sistem. Rezultat proračuna za jednu od osam podsistema interkonekcije prenosnog sistema Srbije prikazan je na slici 3, a sumirani uticaj remonta izabranih dalekovoda na celokupnu razmenu sa susednim sistemima procentualnim smanjenjem (tabela 1).



Slika 3: Vrednosti NTC-a u MW za granicu Bugarska – Srbija

Tabela2. Sumirani uticaj remonta dalekovoda

1. Kosovo B – Skoplje	40.03%
2. Niš - Kosovo B	39.75%
3. Đerdap – Bor	30.69%
4. Niš - Sofija West	15.67%
5. Subotica – Šandorfalva	14.63%
6. S. Mitrovica – Mladost	13.14%
7. Obrenovac – Beograd	9.71%
8. Đerdap – Portile de Fier	8.47%
9. N. Sad – Subotica	2.95%
10. Ernestinovo – S. Mitrovica	0.78%

6. ZAKLJUČAK:

Analizom uticaja remonta dalekovoda na vrednosti prenosnih kapaciteta može da se odredi efekat koji će neraspoloživost nekog elementa imati na sistem. Dobijene vrednosti smanjenja prenosnih kapaciteta od preko 30% pokazuju da taj uticaj nije zanemariv i formiranje liste prema kojem se vrši rangiranje dalekovoda daje informaciju TSO-u, kako optimalno da rasporedi vremenske intervale za remont elemenata prema postojećem stanju sistema. U pojedinim slučajevima TSO može doneti odluku da se ne vrši planirani remont dalekovoda ukoliko se pokaže da je za predviđeni period isplativije da se dalekovod ostavi u pogonu, uz prihvatljiv rizik.

Rezultati proračuna pokazuju da uticaj interkonektivnih dalekovoda ne mora biti dominantan na vrednosti prenosnih kapaciteta i da uticaj unutrašnjih dalekovoda može biti jednak ili čak i veći.

Tokom analize i proračuna prenosnih kapaciteta, postojeća NTC metodologija alokacija prenosnih kapaciteta pokazuje značajne nedostatke: odstupanje programskih od fizičkih veličina u sistemu i neadekvatno praćenje kružnih tokova snaga.

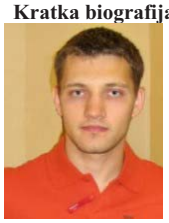
Za operatora prenosnog sistema od interesa je da obezbedi sigurno i pouzdano snabdevanje svih potrošača.

Posmatrajući sa aspekta trgovca interes je samo ekonomski, odnosno izbegavanje situacija zagušenja. Rezultati ovakve analize mogu se iskoristiti na način da poznavanjem uticaja pojedinih dalekovoda i perioda njihovih remonta, trgovci mogu da predvide smanjenje prenosnih kapaciteta u određenom vremenskom periodu na granici koja im je od interesa, i korigovanjem odgovarajućim metodama risk menadžmenta preciznije predvide cene električne energije u bližoj budućnosti.

7. LITERATURA

[1] „Definitions of Transfer Capacities in liberalised Electricity Markets“, ETSO final report , april 2001.
[2] Viktor A. Levi, „Planiranje razvoja elektroenergetskih sistema pomoću računara“, Stylos, Novi Sad 1998.
[3] „Statistical Yearbook“, UCTE, 2005, 2006, 2007.
[4] Davor Bajs, „Ekonomsko–tehnički pristup planiranju razvoja prijenosne mreže“,Magistarski rad,Sveučilište u Zagrebu, 2000.
[5] Zoran Vujasinović, „Upravljanje zagušenjima“,EKC,Bgd,2005.
[6] „Power System Analyser – PSA“, User Manual – basic functions, PSA Team, EKC Belgrade, June 2008.
[7] Miroslav Vuković,Đorđe Dobrijević, Davor Bajs,Goran Majstorović,„Transmission network investment criteria“ – final report, EKC Beograd, Institut Hrvoje Požar, Zagreb,Mart 2007.
[8] „Net Transfer Capacities and Available Transfer Capacities in the internal market of electricity in Europe“, ETSO, 2000.
[9] „Pravila o radu prenosnog sistema“, EMS, April 2008.
[10] „Evaluation of congestion management methods for cross–border transmission“ ETSO, Florence Regulators Meeting, 1999.
[11] „Procedures for cross–border transmission capacity assetment“, ETSO, October, 2001.
[12] „Eksploatacija sistema i koncept dispečerskog upravljanja EES-om“, EKC Beograd, elaborat 1997.
[13] Branko Stojković, Vesna Stojković, „Deregulacija tržišta električne energije i njeni aspekti“, EPCG A.D, Nikšić, 2002.

Kratka biografija:



Andrija Oros *rođen je 6. juna 1983. godine u Vrbasu. Diplomirao je odbranom Master-rada 2008. godine na Fakultetu tehničkih nauka u Novom Sadu, na katedri za elektroenergetiku. Zaposlen u Elektroenergetskom Koordinacionom Centru, Beograd, kao inženjer za studijski rad. Oblast interesovanja: Analiza elekroenergetskih sistema regiona, tržište električne energije.*

NAPOMENA: Ovaj rad je proistekao iz istoimenog diplomskog-master rada čiji mentor je bio profesor dr Ljubomir Gerić.

USLUŽILAC ZA BRZO INDEKSIRANJE DATOTEKA

Bogdan Satarić, Fakultet Tehničkih Nauka, Novi Sad
Dejan Stefanović, Dragan Simić, MicronasNIT, Novosadski Insitut za Informacione Tehnologije, Novi Sad

Sadržaj **– U radu je prikazana realizacija aplikacije za indeksiranje i praćenje liste datoteka u nekoj putanji. Cilj realizacije ovakve aplikacije jeste stvaranje nezavisnog programskog rešenja koje će omogućiti pozadinsko indeksiranje liste datoteka. Na ovaj način rasteretiće se i ubrzati izvršavanje glavne aplikacije koja koristi usluge indeksiranja i praćenja liste datoteka. Date su osnovne karakteristike, kao i kratak opis realizacije date aplikacije.**

Ključne reči **– IPR, JNI, DAEMON**

I. UVOD

Realizacija uslužioca za brzo indeksiranje datoteka je deo šireg projekta Micronas IPR (Intellectual Property Rigths) Editora [2], koji se koristi za evidenciju patenata firme Micronas GmbH. U okviru Micronas IPR Editora [2], postoji mogućnost povezivanja delova teksta sa datotekama preko hiperveza. Na taj način omogućuje se brz pristup datotekama koje su povezane sa datim tekstom. Funkcionalnost povezivanja datoteka sa tekstom u Micronas IPR Editoru [2], zahteva uslugu koja će omogućiti pristup listi datoteka sadržanih u okviru izabranog direktorijuma, ali i onih datoteka koje se nalaze u pod-direktorijumima, nezavisno od broja i dubine propagacije pod-direktorijuma. Imajući u vidu prethodni zahtev i uvažavajući činjenicu da koreni direktorijum C:\ (koji je najopštiji primer indeksiranog direktorijuma), može da sadrži i po nekoliko stotina hiljada datoteka u svojim pod-direktorijumima, potrebno je realizovati pozadinsku aplikaciju koja bi indeksirala i pratila listu datoteka u željenim direktorijumima, i na taj način skratila vreme pronalaženja željene datoteke.

Pojam indeksiranja i praćenja liste datoteka uveden je kako bi se eliminisala potreba za ponovnim čitanjem kompletne liste datoteka u datom direktorijumu i njegovim pod-direktorijumima - svaki put kada bi se zahtevala data lista. Ponovno učitavanje bi bilo neophodno zbog mogućnosti da je od zadnjeg učitavanja liste došlo do brisanja pojedinih datoteka ili dodavanja novih, te bi se lista morala ažurirati.

Umesto toga, obavlja se početno indeksiranje liste datoteka, dok se sve promene u datoj listi registruju kao promene na pojedinačnim datotekama u listi. Na taj način se omogućuje ubrzano ažuriranje promena na elementima liste, a ne na celoj listi. Ovo značajno rasterećuje samu aplikaciju i omogućava bržu dostupnost željenih datoteka.

Uslužilac za brzo indeksiranje datoteka je realizovan kao zasebna aplikacija koja obavlja prethodno opisane funkcije indeksiranja za Micronas IPR Editor [2].

Za proveru validnosti rada uslužioca za brzo indeksiranje datoteka kreirana je posebna aplikacija za brzo i automatsko kreiranje, brisanje i preimenovanje datoteka. Korišćenjem ove aplikacije, jednostavno se može utvrditi tačnost ažuriranja liste datoteka, tj. korektnosti rada uslužioca za brzo indeksiranje datoteka.

Uslužilac za brzo indeksiranje datoteka kao i osnovni program Micronas IPR Editor [2], realizovani su u programskom jeziku JAVA.

II. REALIZACIJA KLASA

Realizacija uslužioca za brzo indeksiranje datoteka u programskom jeziku JAVA omogućila je podelu funkcionalnosti uslužioca na specifične klase i njihove objekte, i na taj način se povećala apstrakcija i lakoća razumevanja nadležnosti pojedinih klasa u okviru samog uslužioca.

A. Klasa IndexMaster

Klasa IndexMaster predstavlja osnovnu klasu aplikacije uslužioca. Ova klasa ima sledeće uloge:

- povezivača komandi, lista i zadataka svih njenih pod-klasa
- pokreće nit klase PathWatch koja je zadužena za praćenje promena na putanjama koje se dodaju u listu
- pokreće nit klase CommandChannel, koja predstavlja apstrakciju komandnog kanala za primanje komandi od strane Micronas IPR Editora [2], i slanja odgovora editoru
- pokreće nit klase IndexMonitor, koja predstavlja raspoređivač zadataka dobijenih na osnovu komandi od osnovne aplikacije (Micronas IPR Editora [2]), kao i zadataka koji se ponavljaju u određenim vremenskim intervalima
- kreira liste objekata klasa IndexMain i IndexInfo koje predstavljaju apstrakciju samih indeksiranih putanja, njihovih stanja kao i informacije o njima

Organizacija klasa i odnosi klasa dati su na slici 1:

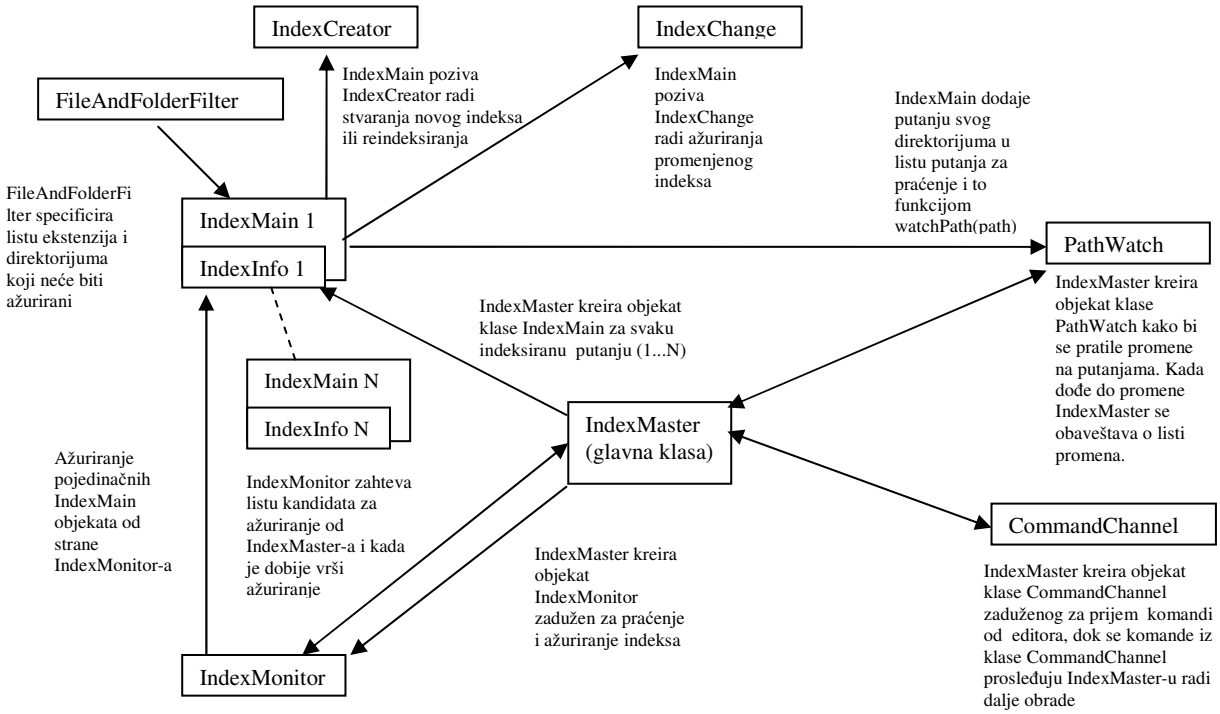
Rad je delimično podržan u okviru projekta TR-6163B Ministarstva za nauku i zaštitu životne sredine Republike Srbije.

Satarić Bogdan, Fakultet tehničkiih nauka u Novom Sadu, Srbija (e-mail: bogdan.sataric@micronasnit.com).

Dejan Stefanović, MicronasNIT, Novosadski Institut za Informacione Tehnologije, Novi Sad, Srbija (e-mail: dejan.stefanovic@micronas.com).

Dragan Simić, MicronasNIT, Novosadski Institut za Informacione Tehnologije, Novi Sad, Srbija (e-mail: dragan.simic@micronas.com).

Napomene:
a) Rad je proistekao iz diplomskog-master rada Bogdana Satarića. Mentor je bio prof.dr Nikola Teslić.
b) Rad je prethodno publikovan na konferenciji TELFOR, Beograd, Novembar 2007.



Sl. 1.Organizacija klasa i njihovi odnosi

B. Klasa PathWatch

Ova klasa predstavlja klasu pratioca putanja odredjenih direktorijuma i promena na datom direktorijumu. Funkcionalnost klase PathWatch je realizovana kao programska nit u kojoj se proverava da li je došlo do promene na određenoj putanji, i ako je došlo do promene – promena se dodaje u listu promena. Data lista se zatim prosleđuje glavnoj klasi IndexMaster i preko nje klasi IndexMonitor (zaduženoj za raspoređivanje zadataka ažuriranja). Promene uključuju brisanje, preimenovanje i dodavanje datoteka i pod-direktorijuma u okviru praćenog direktorijuma.

Same funkcije provere promena na putanjama i preuzimanja zadnje promene su realizovane u programskom jeziku C++. Od datih funkcija je formirana biblioteka filesystemwatcher.dll. Funkcije date biblioteke su uključene u JAVA aplikaciju kao nasleđene (native) funkcije preko Java Native interfejsa (JNI) [1]. Ovakav pristup je potreban iz razloga što je JAVA - programski jezik visokog nivoa apstrakcije, i za sada ne obezbeđuje sve funkcionalnosti niskog nivoa. U ovom slučaju funkcionalnost koja se morala obezbediti van JAVA-e jeste rad sa sistemom datoteka PC računara i registrovanje promena u sistemu datoteka. Radi početka praćenja stanja određene putanje, data putanja se dodaje u listu praćenih putanja funkcijom watchFolder(folderName). Radi provere da li je došlo do promene na određenoj putanji poziva se funkcija watchedFoldersChanged. Poslednja funkcija koja je bitna za praćenje promene stanja putanja jeste funkcija getLastChange, koja vraća zadnju izmenu u okviru direktorijuma, bilo da je u pitanje brisanje, pisanje ili preimenovanje datoteka.

C. Klasa CommandChannel

S obzirom da je uslužioc za brzo indeksiranje datoteka aplikacija čiji rad direktno zavisi od komandi osnovne aplikacije Micronas IPR Editora [2], ukazala se potreba da se na određeni način povežu dve aplikacije, kako bi se obezbedilo pravilno i nesmetano slanje komandi. Rešenje je nađeno u vidu komandne datoteke koja je deljena između dve aplikacije (uslužioca i editora), a u koju se smeštaju komandne poruke i čitaju odgovori uslužioca sa editorske strane, i iz koje se čitaju komandne poruke i šalju odgovori uslužioca sa strane uslužioca. Ovakva implementacija zahteva pažljivo rukovanje pristupom komandnoj datoteci, kako ne bi došlo do istovremenog upisa i čitanja iz date datoteke, tj. kako se ne bi desilo da je datoteka u nekonzistentnom stanju.

Iz tog razloga uvodi se blokirajuća datoteka koja ima zadatak da onemogući istovremen pristup dveju aplikacija (editora i uslužioca) komandnoj datoteci. Princip funkcionisanja je da aplikacija koja prva dobije pristup blokirajućoj datoteci, preuzima pravo pristupa nad komandnom datotekom, i nakon toga upisuje ili čita iz iste. Nakon obavljene operacije čitanja komandi editora ili slanja odgovora uslužioca (u slučaju klase CommandChannel), tj. pisanja komandi editora i čitanja odgovora uslužioca (u slučaju funkcija Editora), blokirajuća datoteka postaje dostupna drugoj strani kako bi mogla da pristupi komandnoj datoteci.

Komande koje Editor može zadati uslužiocu jesu - indeksiranje nove putanje i upit o dostupnosti (postojanju) uslužioca za indeksiranje. Iako se još ne opisuje sama implementacija indeksiranja, potrebno je naglasiti da je indeks predstavljen objektima klasa IndexMain i IndexInfo. Indeks se prvi put kreira za svaki direktorijum

na zahtev Micronas IPR Editora [2], pri izlistavanju sadržaja datog direktorijuma.

D. Klasa IndexMonitor

Ulogu dispečera zadataka indeksiranja, reindeksiranja i ažuriranja indeksa preuzima klasa IndexMonitor. Ova klasa je realizovana kroz programsku nit koja ciklično proverava zahteve editora za indeksiranjem. Takođe, IndexMonitor proverava da li je došlo do izmena u određenim indeksima. Na kraju, zadatak klase IndexMonitor je i da sačuva stanje indeksiranih direktorijuma i to upisom parametara prethodno indeksiranih direktorijuma u posebnu datoteku (po jedna datoteka za datoteke i direktorijume). Parametri su na primer: broj datoteka u direktorijumu, broj pod-direktorijuma, dužina vremena indeksiranja itd.

Ukoliko je došlo do zahteva za indeksiranjem novog direktorijuma od strane editora, IndexMonitor formira nove objekte klasa IndexMain i IndexInfo koji zajedno opisuju datu indeksiranu putanju. Takođe pri indeksiranju, data putanja se dodaje u listu putanja čije izmene se prate, funkcijom addPathToWatchingList(path) klase IndexMaster. Pri indeksiranju pamte se zasebno lista datoteka i lista direktorijuma. U listi direktorijuma nalaze se svi direktorijumi i njihovi pod-direktorijumi. Za svaki direktorijum ili pod-direktorijum u listi pamti se datum zadnje izmene.

Prilikom formiranja novog indeksa takođe se postavlja i vrednost perioda reindeksiranja indeksa. Ovaj vremenski period ne bi trebalo da bude mali pošto bi velika učestanost reindeksiranja opteretila aplikaciju a samim tim i procesor.

Reindeksiranje se obavlja prolaskom kroz listu direktorijuma i pod-direktorijuma i upoređivanjem zapamćenog datuma zadnje izmene, sa datumom nove izmene. Ako su ti datumi različiti (došlo je do modifikacije direktorijuma), obavlja se njegovo reindeksiranje. Reindeksiranje se takođe obavlja periodično, bez obzira na promene u direktorijumima i to u periodu koji je zadan prilikom formiranja indeksa. Ovo je neophodno iz razloga što se na taj način čuvaju podaci o poslednjim izmenama na putanjama, u slučaju da dođe do nepredviđenih događaja tipa - gašenja računara ili gašenja aplikacije.

Objekat IndexMain može biti u stanjima IDLE, INDEXING ili REINDEXING. Data stanja predstavljaju u redosledu: inertno stanje, stanje prilikom indeksiranja i stanje prilikom reindeksiranja putanje. Ova 3 stanja data su u klasi IndexStates.

Sve promene na indeksima registruju se tako što se proveravaju kandidati među postojećim indeksima, i to kandidati za:

- indeksiranje (ako je stanje objekta IndexInfo klase neke indeksirane putanje - "not indexed")
- ažuriranje (ako objekat klase IndexMain date putanje ima elemenata u vektoru tekućih izmena). Izmene se dodaju u ovaj vektor kao posledica registrovanja izmena od strane C++ funkcija klase PathWatch, a predstavljaju brisanje, preimenovanje i dodavanje datoteka.
- reindeksiranje (ako je određeni direktorijum promenjen od prethodnog indeksiranja, tj. ako je neka datoteka u njemu promenjena)

Potrebno je naglasiti da su sve prethodno opisane akcije moguće jedino ako je objekat klase IndexMain (klase koja opisuje indeksiranu putanju) u stanju IDLE. Ovakav zahtev proističe iz činjenice da je jedino neaktivan indeks koji nije u stanjima INDEXING, UPDATING ili REINDEXING, moguće uvesti u neko od ta tri stanja, a ne pre toga. Takođe, u jednom trenutku je moguće izvršavanje dva zadatka simultano, bilo da je u pitanju indeksiranje ili reindeksiranje. Ovakvo ograničenje je postavljeno kako ne bi došlo do preopterećenja procesora sa više istovremenih (re)indeksiranja.

Nakon prethodnih operacija moguće je zapisivanje svih prethodno obrađenih indeksa u posebnu datoteku.

E. Klasa FileAndFolderFilter

Uloga ove klase jeste filtriranje liste datoteka i pod-direktorijuma u okviru indeksiranog direktorijuma. Filtrira se navođenjem liste proširenja datoteka (u konfiguracionoj datoteci) koja će biti izostavljena prilikom indeksiranja direktorijuma.

F. Klasa IndexMain

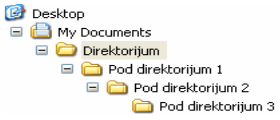
IndexMain predstavlja apstrakciju same indeksirane putanje i kao takva ova klasa je najbitniji deo aplikacije uslužioca. Klasa IndexMain sadrži informacije o putanji koju predstavlja, a informacija se nalazi u objektu druge klase – IndexInfo, iz čega sledi da je objekat klase IndexInfo deo objekta klase IndexMain.

IndexInfo klasa sadrži informacije o putanji, stanju indeksa (indeksiran ili neindeksiran), broju datoteka u indeksu, broju dikrektorijuma u indeksu, vreme indeksiranja, dužini indeksiranja, vremenu reindeksiranja i dužini reindeksiranja.

Sledeći element klase IndexMain jeste vektor tekućih izmena spomenut ranije u radu. On sadrži listu promena na datotekama posmatranog direktorijuma i bitan je za funkciju ažuriranja indeksa. Za ažuriranje indeksa je uvedena nova klasa IndexChange. Metoda doUpdate ove klase omogućuje brisanje datoteka iz liste indeksiranih datoteka (ako je datoteka zaista izbrisana), zatim dodavanja datoteke u listu datoteka (ako je datoteka zaista i dodata) i sekvence brisanja stare datoteke i dodavanja nove preimenovane datoteke u slučaju da je datoteka u stvarnosti preimenovana (ovo eliminiše potrebu za uvođenjem specijalne operacije preimenovanja datoteke u listi).

Bitan korak u ažuriranju indeksa jeste provera da li je zapravo došlo do dodavanja pod-direktorijuma u okviru posmatranog direktorijuma. U ovom slučaju nije dovoljno jednostavno dodavanje samog pod-direktorijuma u listu, pošto je ceo proces indeksiranja vezan za direktno listanje datoteka, a ne direktorijuma (oni se kao takvi ne vide u listi indeksiranih datoteka). Stoga kada je dodat novi pod direktorijum – mora se izvršiti početna operacija indeksiranja tog pod-direktorijuma uz pomoć metode – doIndexing.

Funkcija indeksiranja datoteka je realizovana rekurzivno. Ovakva realizacija je optimalna s obzirom na dubinsku propagaciju pod-direktorijuma u okviru jednog direktorijuma, kao što je prikazano na slici 2:



Sl. 2. Dubinska propagacija pod-direktorijuma

U praksi to znači da će metoda doIndexing dodavati datoteke u listu indeksiranih datoteka, dok će, u slučaju da je u pitanju direktorijum, pozvati samu sebe i istu funkciju ponavljati za datoteke i pod-direktorijume u okviru (pod)direktorijuma pozivaoca funkcije. U toku ažuriranja indeksa objekat IndexMain prelazi iz stanja IDLE u stanje REINDEXING.

Pored ažuriranja, dve moguće operacije jednog indeksa (predstavljenog objektom IndexMain) jesu indeksiranje i reindeksiranje. Indeksiranje se događa samo prilikom prvog stvaranja indeksa direktorijuma na zahtev Micronas IPR Editora [2]. Reindeksiranje se obavlja ako je došlo do modifikacije datoteke i periodično (određeno periodom reindeksiranja). I indeksiranje i reindeksiranje su opisani klasom IndexCreator.

U slučaju indeksiranja indeks prelazi iz stanja IDLE u stanje INDEXING, a u slučaju reindeksiranja u stanje REINDEXING. Suština procesa indeksiranja i reindeksiranja je ista a opisana je prethodno - rekurzivnom funkcijom doIndexing. Indeksirane putanje datoteka se u obliku bajta upisuju u datoteku indeksa. Ovo se čini kako bi se aplikacija učinila nezavisnom od platforme i operativnog sistema, s obzirom da je uslužioc za indeksiranje aplikacija koja će se koristiti ili se već koristi u većem broju računara kompanije Micronas GmbH. Klasa IndexCreator takođe izračunava razliku vremena između početka (re)indeksiranja i kraja (re)indeksiranja.

Po završetku indeksiranja , reindeksiranja ili ažuriranja, obaveštava se glavna klasa IndexMaster, kako bi mogla da se sačuva informacije o indeksu u datoteci. Ove informacije se mogu videti, pošto se uslužioc za indeksiranje pokreće kao pozadinska aplikacija operativnog sistema Windows. Ovo je moguće s obzirom da je JAVA aplikacija uslužioca pretvorena u izvršnu aplikaciju korišćenjem alata Excelsior JET [3]. Pristup uslužiocu za indeksiranje moguć je desnim klikom miša na ikonu u desnom donjem uglu ekrana. Ikona uslužioca je prikazana na slici 3, a zaokružena je crvenom bojom. U okviru menija moguće je izabrati informacije o indeksima, informaciju o verziji uslužioca, a takođe postoji mogućnost isključivanja uslužioca. Meni uslužioca za brzo indeksiranje je prikazan na slici 3:

Sl. 3. Meni uslužioca

III. TESTIRANJE USLUŽIOCA

Testiranje uslužioca za brzo indeksiranje datoteka podrazumeva proveru korektnog indeksiranja, reindeksiranja kao i ažuriranja specificiranih direktorijuma. Radi dobijanja rezultata tačnosti ažuriranja indeksa napravljena je zasebna testna aplikacija RandomFileCreator. Ova aplikacija služi za automatsko generisanje, preimenovanje i brisanje datoteka. Sekvenca prethodnih operacija se zadaje u konfiguracionoj datoteci a ponavlja se ciklično sa novim parametrima preimenovanja kada se završi jedan ciklus operacija. Operacije se obavljaju na svakih 300 milisekundi. Ovo je dovoljna učestanost da bi se mogla proveriti tačnost ažuriranih podataka.

Svako dodavanje, brisanje i preimenovanje datoteka je propraćeno promenom u samoj listi datoteka koja se vidi u okviru editora. Promene se ažuriraju gotovo u realnom vremenu, dovoljno brzo da korisnik editora ima na raspolaganju izmenjenu listu datoteka u svakom trenutku, odmah nakon izmena. Ovo je veoma bitno znajući da direktorijumi koji se indeksiraju za Micronas IPR Editor mogu biti deljeni [2], a samim tim i njihov sadržaj može biti podložan stalnoj promeni i to od strane više korisnika sa različitih računara.

IV. ZAKLJUČAK

Opisana aplikacija predstavlja jednu izvedbu pozadinske aplikacije za praćenje liste datoteka u okviru proizvoljnog broja direktorijuma i pod-direktorijuma. Kao takva ova aplikacija oslobađa glavnu aplikaciju funkcije praćenja indeksa, i na taj način smanjuje opterećenje glavne aplikacije. S obzirom na tu osobinu ovakva aplikacija spada u stalne pozadinske procese, koji se obavljaju s vremena na vreme a služe za ažuriranje određenih informacija. Takve aplikacije se u operativnim sistemima nazivaju i demoni (daemons). Server je preko klase CommandChannel definisao spregu preko koje bi sa njim mogle da komuniciraju i druge aplikacije osim Micronas IPR Edtora [2]. Takođe, uslužilac bi mogao biti smešten na poseban računar, kako bi više klijenata sa različitim aplikacijama mogli da koriste njegove usluge.

LITERATURA

[1] *JNI Technology online training* - http://java.sun.com/
[2] *IPR Editor documentation, Micronas NIT, Novi Sad 2007*
[3] *Excelsior JET Tutorial,* http://www.xlsoft.com/en/index.html

ABSTRACT

This paper presents the application for fast file indexing. The application runs as background application, although it works in collaboration with the editor. As such, this application reduces the workload of editor and holds the latest list of indexed files. The list of indexed files can be passed on to editor (main application) on demand.

FAST FILE INDEXING SERVER

Bogdan Satarić, Dejan Stefanović, Dragan Simić

OTVORENE INOVACIJE KAO MODEL POVEĆANJA KONKURENTNOSTI I TRANSFERA ZNANJA U DRUŠTVU

OPEN INNOVATIONS AS THE MODEL FOR INCREASING COMPETITIVENESS AND KNOWLEDGE TRANSFER IN A SOCIETY

Andrea Katić, Zoran Anišić, Fakultet tehničkih nauka, Novi Sad

Oblast:*INDUSTRIJSKO INŽENJERSTVO I MENADŽMENT*
Kratak sadržaj: *Objašnjen je pojam „otvorenih inovacija" i aktuelno stanje u ovoj oblasti. Detaljnije su obrađeni primeri uspešne primene u akademskom i privrednom okruženju. Predloženi su modeli prihvatljivi i izvodljivi u okruženju. Razmotrene su perspektive i potencijali primene predloženih modela u Srbiji.*
Abstract: *The notion of „open innovations", together with current condition in this field is explained. The examples of successful application in the academic and economic environment are presented in more details. The acceptable and practicable models in the environment are proposed. Perspectives and potential applications of the suggested models in Serbia are considered.*
Ključne reči: *Otvorene inovacije, transfer znanja i tehnologija, mass customization*

1. UVOD

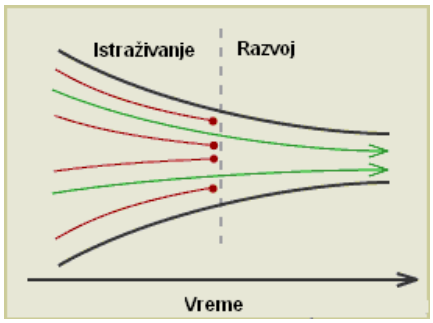
Mnoge činjenice ukazuju da su inovacije glavni pokretač razvoja, uspeha i visoke profitabilnosti kompanija. To znači da se više ni ne postavlja pitanje zašto su inovacije značajne. Umesto toga fokus je na načinu inoviranja i načinu vođenja inovacionog procesa. Novi model upravljanja inovacijama, koji je nazvan *otvoren*, nedavno predstavljen i popularizovan, temelji se na potrebi kompanija da otvore svoje inovacione procese, kao i da kombinuju svoju unutrašnju i spoljašnju razvojnu tehnologiju u cilju stvaranja nove vrednosti.

2. OTVORENE INOVACIJE

Ideju o *otvorenim inovacijama,* prvi put je izložio Chesbrough 2003. godine, a ona je ubrzo privukla pažnju istraživača i konstruktora, i doživela tumačenja i razjašnjavanja u mnogim specijalizovanim časopisima, konferencijama i drugoj tome posvećenoj literaturi.

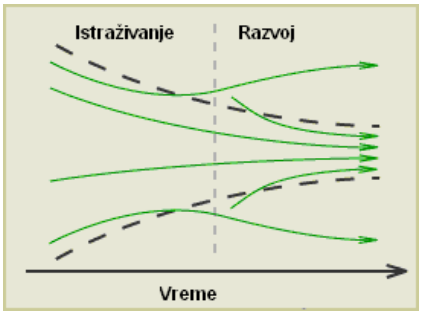
NAPOMENA:
Ovaj rad je proistekao iz diplomskog-master rada Andree Katić „Činioci nacionalnog tehnološkog razvoja i otvorene inovacije kao model povećanja konkurentnosti i transfera znanja u društvu", čiji mentor je bio dr Zoran Anišić, vanr. prof.

U starom modelu *zatvorenih inovacija*, kompanije se oslanjaju na predpostavku da inovativni proces treba da bude kontrolisan od strane kompanije - on se oslanja na samodovoljnost istraživanja (Slika 2.1.). Promene u društvu i industriji, pojava Interneta, koji je približio izvore informacija, dovele su do povećenja mobilnosti eksperata i stvorile nove načine finansiranja kao što su zajednička ulaganja - snaga koja je prouzrokovala rušenje granica u inovativnom procesu. Chesbrough opisuje kako su kompanije u 20. veku mnogo investirale u istraživanje i razvoj (R&D) i zapošljavale najbolje ljude, omogućivši im da razviju najinovativnije ideje i zaštite ih sa strategijama intelektualne svojine (IP). Ostvareni profit je korišćen za ponovno ulaganje u istraživanje i razvoj – u opravdan ciklus inovacija [1]. Chesbrough definiše otvorene inovacije kao:
"naizmenično korišćenje svrsishodnih ulaza i izlaza znanja kako bi se ubrzao interni proces inoviranja, i povećalo tržište za eksternu upotrebu inovacija. Otvorene inovacije predstavljaju paradigmu, koja uverava firme da mogu i trebaju da koriste eksterne ideje isto kao i interne, a zatim interne i eksterne puteve do tržišta, kako bi unapredile svoju tehnologiju." [2].



Slika 2.1.- Zatvorene inovacije

Otvorene inovacije su se pojavile kao model gde firme mogu da komercijalizuju kako sopstvene (interne) tako i tuđe (eksterne) ideje, odnosno tehnologiju i da koriste i unitrašnje i spoljne resurse. U procesu otvorenih inovacija, projekat može da bude lansiran i od strane internog, kao i od eksternog korisnika, pa nova tehnologija može da uđe u raznim etapama procesa (Slika 2.2.).



Slika 2.2. - Otvorene inovacije

Projekat takođe može da se plasira na tržište na razne načine, kao što su out-licencing i spin-off poduhvati kao dodatak tradicionalnim kanalima prodaje. Postoje mnogi načini za upotrebu otvorenih inovacija:

- Povezivanje kupca i prodavca
- Korišćenje inovacija širom industrije
- Kupovina intelektualne svojine
- Investiranje u stvaranje globalnog znanja
- Inovacioni klasteri

Fokus leži u transformaciji pređašnih čvrstih granica kompanije u polupropustljive membrane, kako bi se dozvolilo inovaciji da se lakše kreće između spoljašnjeg okruženja i unutrašnjeg istraživačko-razvojnog (IR) procesa. Glavni pravac inovativnog procesa je takođe i potraga za novim idejama koje imaju tržišni potencijal.

Ideja o otvorenim inovacijama je proizišla iz razvoja open-source softvera gde su primenjeni novi principi vođenja projekta. Istovremeno, ovaj razvoj postao je jedan od najvećih izvora inovacija. To je dovelo do smanjenja troškova, poboljšanja funkcionalnosti i podsticaja za novu industriju. Danas, otvorene inovacije predstavljaju paradigmu, koja povezuje razne pravce menadžmenta. Oblast istraživanja i razvoja se širi u raznim pravcima sa zajedničkim ciljem u povećanju inovativnosti kompanija. Chesbrough takođe identifikuje 5 ključnih tema istraživanja do sada [2]:

- Biznis model – dve važne funkcije, ostvariti dobit unutar lanca dobiti i zadržati deo toga za preduzeće
- Spoljne tehnologije – mogu da osnaže biznis model preduzeća popunjavanjem praznina i stvaranjem komplementarnih proizvoda koji podstiču brže prihvatanje tehnologije
- Teškoće u prepoznavanju, pristupanju i prihvatanju znanja – povećana pažnja na upravljanju znanjem i povezivanju znanja
- Pokretanja (start-ups) – karijere novih tehnologija i istraživanja novih tržišta, i reprezentovanja eksperimenata sa biznis modelima
- IP prava – olakšati izmenu ideja i tehnologija

Međutim, na kraju 20. veka, više faktora se promenilo, kao prvo, mobilnost i raspoloživost visoko obrazovanih ljudi je porasla tokom godina. Kao rezultat toga, velika količina znanja egzistira van istraživačkih laboratorija velikih kompanija. Kada zaposleni menjaju posao, oni sa sobom nose svoje znanje, što dovodi do toga da znanje teče iz firme u firmu. Drugo, dostupnost venture capital-a je nedavno značajno porasla, što omogućava da se dobre i obećavajuće ideje i tehnologije dalje razvijaju van firme, na primer u prduzetničkim firmama. Pored toga, mogućnosti za dalji razvoj ideja i tehnologija van firme, na primer u formi spin-off kompanija ili licenciranja, sve više raste. Na kraju, druge kompanije, koje se nalaze u nabavnom lancu, na primer dobavljači, imaju sve veću ulogu u inovacionom procesu.

Kao rezultat toga, kompanije su počele da tragaju za novim načinima koji će povećati efikasnost i efektivnost njihovog inovacionog procesa. Na primer kroz aktivnu potragu za novim tehnologijama i idejama van firme, a takođe i kroz saradnju sa dobavljačima i konkurentima, sve u svrhu stvaranja vrednosti za kupca. Drugi važan aspekt je dalje razvijanje ili spoljno licenciranje ideja i tehnologija koje se ne uklapaju u strategiju firne (na primer kompanija ASML je Philips-ov spin-off) [3].

Uključivanje drugih delova u toku razvijanja novih proizvoda i tehnologija može da bude od začajne vrednosti. Na primer saradnja sa drugim firmama u sektoru, dobavljačima, univerzitetima, i naravno karajnjim korisnicima.

2.1. Primeri primene otvorenih inovacija

Poslednjih godina pojavio se veliki broj novih rezultata i tehnologija, koje su rezultat otvorenog pristupa kompanija, odnosno otvorenih inovacija. Najviše ih ima iz informacionih tehnologija, odnosno softvera.

2.1.1. Vikipedija (Wikipedia)

Vikipedija je enciklopedija slobodnog sadržaja na Internetu koju razvijaju dobrovoljci uz pomoć viki softvera. Članke na Vikipediji može menjati svako sa pristupom internetu. [4]. Projekat je počeo 15. januara 2001. godine kao podrška Nupediji, enciklopediji koju su pisali eksperti. Sada njome rukovodi neprofitna Zadužbina Vikimedija. Vikipedija ima više od 7 miliona članaka na 253 jeziku, od čega je više od 2,1 miliona članaka na verziji pisanoj engleskim jezikom i preko 57.000 članaka napisanih srpskim jezikom (podaci su od decembra 2007) [4]. Vikipedijine članke zajednički pišu dobrovoljci širom sveta, a veliku većinu članaka može da uređuju svako ko ima pristup internetu. Kako je popularnost Vikipedije konstantno rasla od njenog početka, ona se sada nalazi među 10 najposećenijih sajtova.

Vikipedijini članci su dostupni pod Licencom za slobodnu dokumentaciju GNU-a (GLSD). Ova licenca je danas jedna od mnogih „kopileft“ licenci koje omogućavaju umnožavanje, stvaranje izvedenih radova, kao i komercijalno korišćenje sadržaja sve dok se naznačuju autori dela i sve dok rad i radovi ostaju pod istom licencom. Kada autor da svoj originalni materijal projektu, autorsko pravo ostaje njemu, ali on se slaže da ga daje dostupnim pod GLSD-om. Tako materijal sa Vikipedije može biti inkorporiran u ona dela koja takođe koriste ovu licencu. [4]

2.1.2. SAP - Systems, Applications and Products in Data Proccesing

Potrebu za univerzalnim poslovnim softverskim rešenjem, koje bi objedinilo i unapredilo poslovanje, prepoznala je još sedamdesetih godina grupa inženjera iz Manhajma i osnovala SAP - Systems, Applications and Products in Data Proccesing. Od tada, tokom više od trideset godina rada, SAP je prešao dug put od mainframe, client-server, do e-business generacije i uvek igrao pionirsku ulogu u primeni najsavremenijih rešenja na polju informacionih tehnologija.

Danas je SAP, sa sedištem u Valdorfu u Nemačkoj, vodeći svetski ponuđač poslovnih softverskih rešenja. Više od 24.450 korisnika u preko 120 zemalja sveta danas pokreće više od 84.000 instalacija SAP softvera - od specifičnih rešenja upućenih malim i srednjim preduzećima do enterprajz

nivoa poslovnih paketa za globalne organizacije. SAP rešenja su otvorena i fleksibilna, podržavaju baze podataka, aplikacije, operativne sisteme i hardvere skoro svih proizvođača.

SAP Business One i mySAP All-in-One rešenja za mala i srednja preduzeća podržavaju specifične zahteve ovih preduzeća, a ono što ih izdvaja jeste korišćenje otvorene tehnologije, koja dozvoljava praktično neograničeno širenje ovih sistema kako se i obim poslovanja bude širio, mogućnost "da rastu" zajedno sa preduzećem.

2.2.3. Google My Maps

Google je američka javna korporacija, specijalizovana za Internet pretragu i reklamiranje na Internetu. Sedište kompanije je u mestu Mauntin Vju, Kalifornija [4]. Guglov poslovni cilj je „da organizuje svetske informacije i učini ih svetski dostupnim i korisnim".

Google mape je besplatna web usluga pregleda mapa koja je podržana aplikacijama i tehnologijom koju nudi Googl. Ona nudi mape ulica, planer ruta za bicikle, pešačke staze i vozila, kao i urbani biznis lokator za brojne zemlje širom sveta. U aprilu 2007, Google je dodao novu poguslugu Moje mape. Ona omogućava korisnicima i poslovnim subjektima izradu vlastitih mapa sa pozicioniranjem značajnih stvari markerima, kao i crtanje poligona. Karte koje su modifikovane koristeći Moje karte se mogu sačuvati za kasnije pregledavanje i takođe se mogu objaviti (ili označiti kao privatne).

2.2.4. Yahoo – Answers

Yahoo je poznati svetski internet pretraživač, koji mnogi smatraju najvećom konkurencijom Googl-u. Yahoo su 1994. osnovali David Filo i Jari Yung . Smatra se da Yahoo ima najpoznatiji besplatan veb mejl. Yahoo! Answers predstavlja veb stranicu tržišta znanja koju je Yahoo! lansirao dana 2005. Ona omogućava korisnicima da postavljaju pitanja kao i da daju odgovore na pitanja drugih korisnika. Sajt daje priliku članovima da zarade bodove koji služe kao način podsticanja učešća. Decembra 2006, servis je imao 60 miliona korisnika i 65 miliona datih odgovora. Yahoo Answers je tako postala druga najpopularnija Internet stranica koja se koristi kao referenca nakon Vikipedije, prema comScore-u. Zadovoljni korisnici ističu kako im je ovaj servis omogućio značajne uštede novca i vremena ali i dao ideje čija je vrednost neprocenjiva.

2.3. Masovna proizvodnja po željama kupca *(Mass Customization)*

Mass Customization proizašla je iz modela otvorenih inovacija i danas predstavlja novi trend u proizvodnji, koji komponuje proizvode po željama kupaca. Ovaj pristup u današnje vreme sve više dobija na popularnosti usled rastućeg broja varijanti prizvoda i povećanih mogućnosti za e-trgovinom [5]. Konkurentnost na globalnom tržištu zahteva od kompanija da promene dosadašnje pristupe u proizvodnji, koji su se oslanjali na „gledište prodavca", u pristupe koji će biti okrenuti kupcu i njegovim željama. Rezultat toga je drastično povećenje broja varijanti proizvoda i troškova. Kako bi zadržale visoku konkurentnost na tržištu, kompanije vrše modulovanje svojih proizvoda, a transfer od proizvoda, gde kupci nisu uključeni, do modularnih proizvoda, koji uključuju individualne želje kupaca, predstavlja jednu od najvažnijih industrijskih strategija današnjice. Razvoj IT sektora je omogućio stvaranje takvih softvera, koji podržavaju proces kreiranja proizvoda po željama kupaca. Pomoću modula, koji kupci sami biraju, ovi softveri komponuju proizvod koji je korisnik zamislio.

3. MOGUĆNOSTI PRIMENE MODELA OTVORENIH INOVACIJA U DOMAĆOJ PRAKSI

3.1. Primer za MSP-Softver za konfigurisanje prizvoda i usluga u virtuelnoj proizvodnji

Ideja projekta se odnosi na mala i srednja preduzeća, koja proizvode proizvode prilagođene željama kupaca dok istovremeno održavaju troškove i vreme isporuka niskim odnosno kratkim po pristupu *mass customization-a* [6]. Međutim, većina malih i srednjih preduzeća sa proizvodnim platformama ne mogu da priušte konstruisanje sopstvenog konfiguratora proizvoda. Zbog toga takva preduzeća imaju veliku potrebu za više ekonomičnim i fleksibilnijim konfiguratorom proizvoda. Glavni cilj je da se napravi modularni, lako podesiv, standardni softver i alat za standardne usluge koji služi za čuvanje i sakupljanje znanja, kao i za implementaciju proizvod konfiguratora u malim i srednjim preduzećim koja proizvode proizvode po *mass customization* principu.

Mnoge kompanije same razvijaju sopstvene konfiguratore proizvoda, tako se pravila potrebna za kombinovanje komponenti proizvoda, kao i dizajn celog proizvoda statički implementiraju u sistem za konfigurisanje. Kao posledica toga, svaka promena proizvoda i/ili nadogradnja i tako inicirana promena, zahteva celokupnu promenu u izvornom kodu sistema za konfigurisanje. Direktno pretvaranje pravila u izvorni kod i dovodi do sve veće zavisnosti od softverskih inženjera, jer ostali korisnici ne mogu uređivati ili menjati konfiguracijska pravila. U tom kontekstu, potreba za oslobađanjem *know how*-a koje bi došlo do eksternih inženjera i komjuterskih naučnika predstavlja veliku psihološku prepreku za mnoge kompanije. Pored toga, opisan pristup dovodi do stvaranje krajnje nejasne baze pravila, a koja je, nakon nekog vremena, jedva upotrebljiva.

Samo softver inženjeri mogu unaprediti softver proizvod konfiguratora koji je kompanija sama razvila. Izvorni kod je izuzetno kompleksan i nejasan zbog velikog broja postojećih konfiguracijskih pravila. Cilj projekta bio je razviti okvir (*framework*) za stvaranje pojedinačnih kofiguratora proizvoda rešenja za mala i srednja preduzeća u mehaničkom i mašinskom inženjerstvu. Konačni proizvod se sastoji od softvera (*framework a*) i standardizovonih komponenti usluga, uz pomoć kojih proizvođači mogu razviti specifične siteme za konfiguraciju.

Razvijena rešenja za konfiguratore proizvoda poseduju jedinstvene karakteristike kao što su:

- virtuelni 3D-model proizvoda koji nastaje kao rezultat procesa konfiguracije, koji se direktno primjenjuje u daljnjem procesu razvoju proizvoda
- prateće standardizovane usluge za podršku uvođenja sistema u kompanije
- preciznu procena odnosa troškova i dobiti još pre početka projekta s ciljem da se minimizira rizik
- elementarni softver sistem, koji se može prilagoditi klijentima
- strukturu otvorenih komunikacija između uključenih sistema i koncentracija na relevantne objekate i atribute u konfiguraciji, što olakšava održavanje inženjerima i dovodi do manje zavisnosti od kompjuterskih naučnika

3.2. Primena na univerzitetu-Program „Aktivan Student“

Softverski program „Aktivan Student“, koji funkcioniše po modelu otvorenih inovacija, omogućava međusobnu interakciju na relaciji student-student, student-profesor i preduzeće-profesor-student. Program podržava više funkcija.

3.2.1.“Pitaj i odgovori“:

Ova funkcija, izgrađena na principu „otvorenosti“, pruža mogućnost studentima da postavljaju pitanja, kao i da daju odgovore na pitanja drugih studenata, stvarajući tako bazu znanja o datoj oblasti dostupnu svima. Istražujući obaveznu ili drugu raspolozivu literaturu vezanu za tekuće kurseve, domaće zadatke, seminarske radove, projekte i slično, mnogi studenti nailaze na razne nepoznanice, susreću se sa dilemama i nejasnoćama o kojima žele da prodiskutuju ili postave pitanja. Studenti, koji poseduju znanje o datom problemu, mogu pružiti odgovor na postavljeno pitanje, dati neka uputstva ili pojasniti samo pitanje i tako svoje znanje podeliti i pomoći drugima. Na ovaj način baza znanja svekodnevno raste i stvaraju se nove informacije dostupne svima. Pretraživanjem ove baze korisnici mogu brzo i lako da dođu do vrednih informacija o aktuelnoj oblasti.

3.2.2. Aktuelni projekti

U ovoj funkciji profesor je u mogućnosti da postavi određene zadatke, probleme i projekte, studentima koji žele da učestvuju u nastavnim ili vannastavnim aktivnostima. Profesor postavlja temu zadatka, odnosno projekta, kao i niz podpitanja, koja su neophodna za njegovu realizaciju. Studenti resavaju postavljene zadatke samostalno, u timu ili u saradnji sa nekim preduzećem, koje je predmet rešavanja zadatka. U toku rada, otvara se poseban nalog za formirani tim ili grupu. Na ovaj način studenti prezentuju svoje ideje, koje stalno nadograđuju jedna drugu i zajedničkim naporima dolaze do rešenja zadatka. Dok se uloga profesora sastoji u kordinaciji, davanju smernica odnosno vođenju projekta.

3.2.3. Baza podataka sa literaturom i završnim projektima

Studenti i profesori su u mogućnosti da postave odgovarajuću literaturu vezanu za tematiku predmeta. Takođe svi završeni projekti su pristupačni svim zainteresovanim licima koji ih u svakom trenutku mogu pregledati i na taj način doći do vrednih ideja i informacija. Pored ovoga u bazi podataka nalaze se sve i ideje koje nisu iskorišćene i komercijalizovane te u svakom trenutku mogu biti ponovo pokrenute i naći svoje tržište.

3.2.4. Sve o ispitu

Ovaj deo spada u sad već standardne funkcije vezane za rad studenata na nekom univerzitetu. U ovom slučaju, postavljen je radi kompletnosti „paketa“ i radi integracije svih funkcija sistema u jedinstven program. Radi se o servisu, koji daje studentu sve relevantne podatke o ispitu, bilo položenom, bilo da je u pitanju predstojeci.

3.2.5. Praksa

Ova strana daje informacije o preduzećima, koja su zainteresovana da prime studente na stručnu praksu u intervalu od 2-6 nedelja u zavisnosti da li se radi o osnovnim ili diplomskim-master studijama. Daju se osnovne informacije o preduzeću, tipu posla, opis dnevnih zadataka, mogucnosti usavršavanja, kao i mogućnost dalja perspektive po zavrsetku studija.

4. ZAKLJUČAK

Otvorene inovacije su se pojavile kao model gde firme mogu da komercijalizuju kako sopstvene (interne) tako i tuđe (eksterne) ideje, odnosno tehnologiju i da koriste i unutrašnje i spoljne resurse. U procesu otvorenih inovacija, projekat može da bude lansiran i od strane internog, kao i od eksternog korisnika, pa nova tehnologija može da uđe u raznim etapama procesa.

Pojava modela otvorenih inovacija predstavlja šansu za Srbiju da istakne prednosti kao što su kvalitet obrazovnog sistema i raspoloživost naučno-obrazovnog kadra, a ublaži mane koje se odnose na ulaganje u obrazovanje, kao i ulaganje u istraživanje i razvoj. Putem modela otvorenih inovacija kompanije su u stanju da za karće vreme i sa manje finansijskih sredstava dođu do novih rešenja. Takođe, novi model u poslovanju pod nazivom *mass customization*, koji predsatvlja masovnu prizvodnju po želji kupaca, predstavlja drugi bitan trend u inoviranju koji domaće kompanije treba da isprate.

5. LITERATURA

[1] H.W.Chesbrough, „The era of open innovation“, MIT Sloan Management Review, Vol.44, No.3, 2003, pp.35-41.

[2] H.W.Chesbrough, „New puzzles and new findings“, In H.W.Chesbrough & W.Vanhaverbeke & J.West (Eds.), Open innovation: Researching a new paradigm: pp.15-33, Oxford University Press, Oxford, 2006.

[3] *http://www.asml.com*

[4] Vikipedija**,** *www.wikipedia.org*

[5] I.Furstner, Z.Anišić, I.Ćosić, „Mass Customizatiom: trends, research and application“, XIV Međunarodna naučna konferencija Industrijski sistemi - IS 2008, Novi Sad, okt. 2008., pp. 493-498.

[6] M.Abramovici, M.Ghoffrani, M.Neubach, S.Bertram, „KoViP - Product Configuration Software and Services for Virtual Products“, Proc. of Joint Conference IMCM-07 & PETO-07, 21.-22.06.07, Hamburg, Gito Verlag, Berlin, 2007, ISBN: 978-3-940019-03-5, S. 1-12

Kratka biografija:

Andrea Katić rođena je u Novom Sadu 1984. god., gde je završila osnovnu školu i gimnaziju »Isidora Sekulić« 2003. god.. Diplomski-master rad na Fakultetu tehničkih nauka iz oblasti Inženjerski menadžment odbranila je 2008.god.

Dr Zoran Anišić, van.prof., rođen je u Subotici 1969.god. Na Fakultetu tehničkih nauka doktorirao je 2002. god., a od 2008. je u zvanju vanrednog profesora za užu naučnu oblast Projektovanje proizvodnih sistema, organizacija i menadžment.