

METHODOLOGY ENHANCEMENTS OF THE EDUCATIONAL PROCESS IN DSP PROGRAMMING COURSE

UNAPREĐENJE METODOLOGIJA EDUKACIONOG PROCESA U KURSU DSP PROGRAMIRANJA

Anita SABO

Visoka tehnička škola strukovnih studija – Subotica Tech

Address: Marka Oreškovića 16, 24000 Subotica, Srbija

Phone: +381-24-655-243

E-mail: saboanita@gmail.com

Bojan KULJIĆ

Visoka tehnička škola strukovnih studija – Subotica Tech

Address: Marka Oreškovića 16, 24000 Subotica, Srbija

Phone: +381-24-655-271

E-mail: bojan.kuljic@gmail.com

Tibor SZAKÁLL

Visoka tehnička škola strukovnih studija – Subotica Tech

Address: Marka Oreškovića 16, 24000 Subotica, Srbija

Phone: +381-24-655-246

E-mail: szakall.tibor@gmail.com

UNAPREĐENJE METODOLOGIJA EDUKACIONOG PROCESA U KURSU DSP PROGRAMIRANJA

METHODOLOGY ENHANCEMENTS OF THE EDUCATIONAL PROCESS IN DSP PROGRAMMING COURSE

Keywords:

educational tool, simulation of DSP filters, interactive framework

Abstract:

This paper presents a practical solution for interaction between students and software/hardware systems in education. The development of science and the technology allowed significant enhancements to the methods in presentation of new materials to students. The main purpose of this research was for the students to gain some basic insight into elementary techniques needed for design, implementation and merging of hardware and software components used in testing of the algorithms for digital signal processing in real time. Main goal was to develop Digital Signal Blocks (DSP) which would be used as intuitive bridge between students and DSP systems by eliminating obscure software interface. Over the years it was established that students waste the majority of their time in classes on adopting software interface which leaves them with very little time to master the key principles in the curriculum. This is especially true in the case of the embedded systems because there students must learn to work with several software packets. The developed software system allows students to concentrate on just one graphical interface which is intended to handle every aspect of embedded system design – from algorithm simulation to the hardware implementation. Through the work with the students it was concluded that the developed DSP blocks presented very good assistance in educational process. Since real hardware systems were used in this case noise was introduced in the system which does not exist in simulation software and therefore this option produced much larger capabilities for development of the robust algorithms. The advantage of this approach lies in the fact that the anchor point in education was transferred from the mere observation of the lecture presentation to the active participation in the aggregation of the new material.

Ključne reči:

alati u edukaciji, simulacija DSP filtara, interaktivni framework

Sažetak:

U ovom radu je predstavljeno praktično rešenje za interakciju studenata sa softverskim-hardverskim sistemima u edukaciji. Razvoj nauke i tehnologije je omogućio značajne pomake u unapređenju metoda prezentovanja gradiva studentima. Glavni cilj ovog rada je bio da se omogući studentima da ostvare dublji uvid u elementarne tehnike potrebne za dizajniranje, implementaciju i ostvarivanje sinergije između softverskih i hardverskih komponenti korišćenih u testiranju algoritama za digitalnu obradu signala u realnom vremenu. Naglasak je stavljen na razvoj digitalnih signalnih blokova koji služe kao intuitivna sprega između studenata i DSP sistema putem eliminacije bilo kakvog komplikovanog interfejsa. Tokom godina je zaključeno da studenti gube većinu vremena na savladavanje korisničkog interfejsa nakon čega im ostaje mnogo manje vremena za savladavanje ključnih delova gradiva. Ovo pogotovo važi u slučaju ugrađenih hardverskih sistema iz razloga što je potrebno savladati više različitih softverskih paketa. Razvijeni softverski sistem omogućava studentima da se koncentrišu samo na jedan grafički interfejs koji je projektovan da adresira sve aspekte projektovanja ugrađenih sistema – od simulacije algoritama do hardverske implementacije. Kroz rad sa studentima je zaključeno da razvijeni DSP blokovi predstavljaju veoma dobru ispomoć u edukacionom procesu. Pošto su se za izvođenje nastave koristili fizički hardverski sistemi u testiranje su bile uključene i smetnje iz stvarnog sveta što je rezultiralo dobijanjem robusnih algoritama. Glavna prednost ovakvog pristupa leži u činjenici da je težište u edukacionom procesu premešteno iz pukog prisustva prezentaciji gradiva na aktivno učešće u savladavanju nove materije.

UVODNO RAZMATRANJE / INTRODUCTION

Trenutno dostupni razvojni DSP sistemi ne zadovoljavaju u potpunosti današnje stroge i robusne zahteve, posebno u svetlu drugačijeg profila studenata novih generacija. Na polju digitalne obrade signala, primenjene metode ne mogu zadovoljiti sve veće i kompleksnije zadatke. Iz ovih razloga je potrebno razviti nove metode, gde se stavlja naglasak na sve brži razvoj i na brže rešavanje problema. Ovde je predstavljeno rešenje u vidu izgradnje interaktivnog sistema, gde je podržan razvoj na višem programerskom nivou.

Metoda interaktivnog razvoja u realnom vremenu u digitalnoj obradi ima brojne prednosti kod razvoja aplikacije. Ubrzava se razvojni tok baš iz razloga što je u ranoj fazi moguće testiranje algoritma filtra, poređenje sa realnim fizičkim DSP sistemima, kao i korekcija eventualnih nedostataka. Integracija u već postojeći razvojni alat (MATLAB) olakšava primenu ove razvojne metode, a njeno integrisanje u jednostavno grafičko okruženje omogućuje koncentrisanje na probleme i algoritme DSP umesto na sam programski jezik.

Pomoću softverskih komponenti se implementacija dešava na višem nivou bazirajući se na već postojećim komponentama. Arhitektura softverskog okruženja dolazi do stvarnog izražaja kod izgradnje specifikacije za implementaciju algoritma. Implementacija se vrši na osnovu blok dijagrama samog algoritama. Primenom softverskog okruženja, programer se udaljava od klasičnog programiranja koje je blisko hardveru, razvijen softver se izvršava u softverskom okruženju potpuno nezavisno od hardvera.

Veoma bitna stvar kod svake tehnologije je standardizacija. To znači uvođenje protokola za komunikaciju. Definisanjem ovakvog protokola za sve osnovne DSP komponente razmena informacija između DSP komponenti u MATLAB-u bi postala veoma uprošćena i jednostavna, čime bi došlo do značajnog smanjenja grešaka pri konstruisanju složenih DSP komponenti od strane korisnika.

Pošto većina ljudi informacije najlakše tumači ako su prezentovane u vizuelnoj formi logično je bilo razviti alate za grafički prikaz koji bi obuhvatali najvažnije oblasti kao što su: kontroler, protok podataka i procesiranje podataka.

CILJEVI I METODE / OBJECTIVES AND METHODS

Komercijalno dostupni softverski paketi iz DSP oblasti podrazumevaju da korisnik već raspolaže određenim znanjem, a ne da pokušava da usvoji osnovne principe iz neke oblasti. Pored ovoga proizvođači, kao npr. Texas Instruments, izdaju softver koji se može koristiti

samo sa određenom familijom procesora i ne mogu se upotrebiti za razmatranje uopštenih slučajeva. Tokom primene softverskih paketa MATLAB i Code Composer Studio na Visokoj tehničkoj školi strukovnih studija u Subotici primećeno je da studenti imaju više problema pri navikavanju na rad u programskim paketima nego sa usvajanjem osnovnih teorijskih principa iz razmatranih oblasti obrade signala.

Stvaranjem jednog DSL opisnog jezika oblast se može razdvojiti na projektovanje specifičnog dela (obrada signala) i dela za realizaciju (hardver, softver). Opisni jezik omogućava da stručnjaci iz date oblasti razvijaju samo algoritme. Prednost ovog načina razvoja je da se za kraće vreme razvijaju kvalitetni algoritmi i robusna realizacija.

Polazeći od hipoteze istraživanja da je razmatranje, razvoj i testiranje algoritama za filtriranje niskofrekventnih signala u realnom vremenu uglavnom rađeno u simulacionom okruženju, treba naglasiti da rezultati simulacije po pravilu ne sadrže karakteristične granice realnog sistema. Zato je potrebna realizacija takvog sistema koji obezbeđuje efektivno testiranje algoritama za digitalnu obradu niskofrekventnih signala u realnom okruženju i realnom vremenu.

Upravo je ovde primećen problem u edukaciji određene kategorije studenata. Naime, studenti gotovo da nemaju dodira sa praktičnim analizama i testiranjima algoritama za obradu signala na fizičkim platformama, pa se stoga javlja potreba za uvođenjem sistema koji bi objedinio i simulacione i praktične alate kako bi se ostvarilo što intuitivnije i po mogućnosti interaktivno usvajanje znanja kod studenata.

Pri radu na fizičkim DSP platformama javlja se šum koji sam hardver unosi u signal. Pod time se podrazumeva šum u analognom delu štampane ploče i šum koji unosi sam A/D konvertor. U zavisnosti od korišćenih komponenti može se javiti šum različite raspodele (beli šum, šum usled neadekvatnog povezivanja sa audio opremom itd.). Takođe kvalitet komponenti i kvalitet izrade štampane ploče ima presudnu ulogu kod nivoa prisutnog šuma. Zbog ovakvog „realnog“ okruženja simulacioni softveri ne mogu da u potpunosti prikažu uticaj šuma na korisni signal. Takođe veliki problem kod simuliranog šuma leži u činjenici da je šum koji će se javiti na fizičkoj štampanoj pločici nepoznat, tako da bi razvijeni algoritam mogao da bude neefikasan ako je izvršena pogrešna pretpostavka o vrsti i nivou šuma. Robustan algoritam je onaj čija je funkcija neosetljiva u odnosu na šumove iz određenog unapred zadanog opsega vrednosti.

Upravo iz tih razloga je postavljen cilj interaktivnog načina projektovanja filtera za digitalnu obradu niskofrekventnih signala u radu sa studentima. Interaktivni rad podrazumeva zadavanje ulaznog signala u razvijeni algoritam i posmatranje odziva koji daje

hardverska DSP platforma. Na taj način studenti dobijaju direktnu informaciju o uticaju algoritma na signal kao i o nivou i obliku šuma na testiranoj platformi.

Cilj istraživanja je projektovanje specifičnog programskog jezika za digitalnu obradu signala, čija bi primarna namena bila efikasno upućivanje studenata u osnovne koncepte digitalne obrade signala na stvarnim hardverskim platformama. Da bi se obezbedila maksimalna upotrebljivost projektovanog opisnog jezika korišćene su metode objektno orijentisanog programiranja kao i grafički prikaz osnovnih gradivnih struktura koje se koriste u projektovanju krajnjeg sistema. Izabrani grafički elementi opisnog jezika u ogromnoj meri olakšavaju projektovanje algoritama i u isto vreme značajno smanjuju mogućnost nastajanja greške pri projektovanju. Još jedna velika prednost grafičkog prikaza elemenata se ogleda u lakoći spajanja jednostavnijih elemenata u kompleksne sisteme u odnosu na klasične programske metode.

Razvijeni sistem predstavlja spregu hardvera i softvera čija je primarna namena unapređenje usvajanja znanja i veština kod studenata pri proračunu i implementaciji DSP rešenja u digitalnoj obradi signala, pa je radi lakšeg razumevanja ciljeva rada prikazana funkcionalna šema projektovanog sistema kroz primer izvođenja rada sa studentima, slika 1. Pored toga, dat je ukratko opis jedne laboratorijske vežbe iz predmeta "Obrada zvuka". Cilj vežbe je da se studentima demonstrira delovanje projektovanog filtra na audio signal. Vežba je podeljena na pet delova:

1. Projektovanje filtra sa unapred zadatim parametrima.
2. Testiranje uticaja filtra na audio signal u simulatoru (audio signal je obezbeđen u vidu unapred generisanog binarnog fajla).
3. Testiranje uticaja filtra na audio signal u koji je unešen šum u simulatoru (signal koji predstavlja mešavinu audio signala i šuma je obezbeđen u vidu unapred generisanog fajla).
4. Testiranje uticaja filtra na audio signal na fizičkoj DSP platformi (audio signal je obezbeđen preko audio generatora koji je priključen na audio ulaz DSP razvojne ploče).
5. Testiranje uticaja filtra na audio signal na fizičkoj DSP platformi u koji je uključen šum (audio signal je obezbeđen preko audio generatora koji je priključen na audio ulaz DSP razvojne ploče, a šum je obezbeđen preko generatora belog šuma koji je priključen na drugi audio ulaz DSP razvojne ploče; mešanje audio signala i šuma se vrši unutar samog DSP procesora nakon A/D konverzije (Analog to Digital Conversion)).

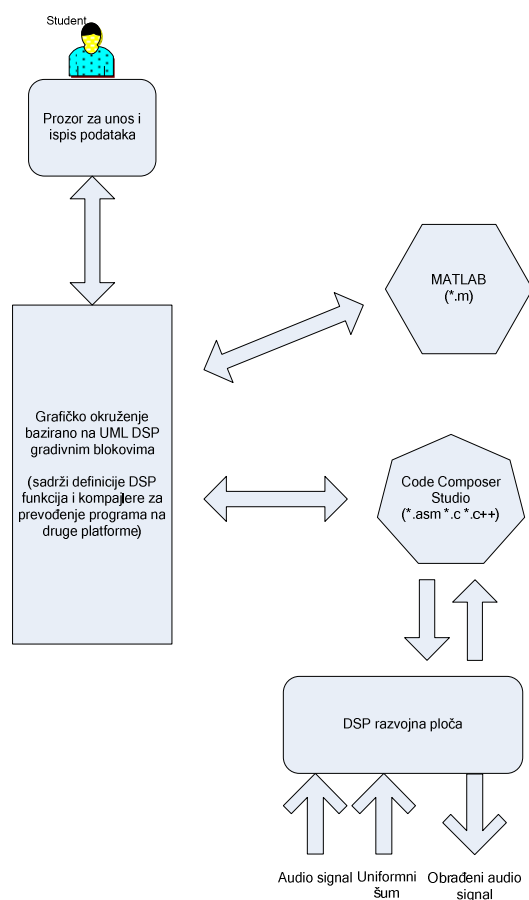
Glavna prednost ovakvog koncepta rada leži u činjenici da se student koncentriše samo na jedan prilagođen programski alat, pa ne mora da odvajati vreme na savladavanje više programskih paketa nego se može koncentrisati na usvajanje demonstriranih principa iz nastavnog gradiva. Na ovaj način smanjuje se opterećenje i studenta i predavača čime se podiže efikasnost u nastavi, što je od posebnog značaja kada je reč o relativno malom obimu nastave i ograničenom predznanju studenata koji treba da savladaju samo osnovne principe i koncepte obrade signala.

Pri obrazovanju u oblasti DSP veoma je bitna mogućnost komparacije rezultata simulacije i rezultata dobijenih na fizičkom hardveru. Tako je moguće odmah steći praktično iskustvo vezano za implementaciju teorijskih proračuna na realnim DSP sistemima (npr. razlika u rezultatu ako se primenjuje 8 bitna aritmetika ili 32 bitna aritmetika za realizaciju koeficijenata filtra). Tako studenti stiču dublji uvid u principe praktičnog funkcionisanja algoritama za obradu signala jer pri izvođenju proračunatih formula u simulacionim softverskim paketima softver koristi 32 bitnu ili čak 64 bitnu preciznost sa pokretnim zarezmom, pa se ne mogu lako uočiti problemi ograničene preciznosti na realnim DSP sistemima. Takođe, pri radu sa fizičkim DSP procesorima uglavnom su na raspolaganju 16 bitni registri koji podržavaju samo celobrojne matematičke operacije tako da studenti mogu da vide direktan uticaj zaokruživanja rezultata matematičkih operacija primenjenih na digitalni signal.

Razvijeni softver razvijen podržava i mogućnost smanjivanja rezolucije. Iako DSP hardver podržava npr. 16 bitne operacije, softver može da koristi samo prvih 8 bita čime se od 16 bitnog procesora veštački stvara 8 bitni. Na ovaj način studentima se pruža mogućnost da lako izvrše uporedni pregled funkcije odziva, polova, nula itd, za projektovane filtre u 8, 12, 16 i 32 bitnoj rezoluciji. Ovakav vid direktne edukacije osposobljava studente da predvide i praktično provere ponašanje proračunatih filtara na fizičkom hardveru čija preciznost u izvođenju matematičkih operacija je daleko niža od one koja se može ostvariti na simulacionom softveru na personalnim računarima.

Kao što se iz prethodnog teksta može videti cilj projektovanog sistema nije bio da zameni postojeće široko rasprostranjene programske pakete, nego da omogući korišćenje više programskih paketa kroz objedinjeni grafički interfejs čiji je osnovni zadatak da generiše određene DSP algoritme i prikazuje rezultate primene tih algoritama u simulacionim i fizičkim okruženjima sa audio i video signalima.

Slika 1: Generalni pregled realizovanog sistema



Pregledom teksta vežbe mogu se izvući zaključci o izvođenju edukativnog procesa. Na prvi pogled se vidi da u nekim segmentima dolazi do drastičnog odstupanja od tradicionalnih metoda predavanja. Naime pri pisanju stručne literature uvek se stavlja akcenat na sažetost, konciznost i preciznost izložene materije. Autori ovog rada se u potpunosti slažu sa takvom praksom koja se u mnogobrojnoj literaturi zalaže za ovakav pristup izlaganja materije, ali samo u slučaju da je ta literatura namenjena čitaocima koji su već upućeni u osnove posmatrane naučne oblasti i sada teže da unaprede svoje postojeće znanje. U komunikaciji sa studentima utvrđeno je da su osobine kao što su sažetost, neponavljanje i ravnomerna zastupljenost u izloženom materijalu vrlo često veoma štetne po pitanju brzine usvajanja gradiva. Jedan od velikih problema na koji su se studenti žalili je velika teškoća da se odrede bitni segmenti gradiva koji predstavljaju osnovu koju prvo treba identifikovati i usvojiti, a tek nakon toga preći na razumevanje ostatka gradiva. Naime ako se dosledno poštuju principi sažetosti i konciznosti tada će, da ne bi bilo “nepotrebnog” ponavljanja, i najvažnija osobina i najmanje važna osobina dobiti otprilike istu količinu prostora u materiji. Ovo je čak poželjno kada čitalac ima dovoljnu količinu predznanja da ne

bi došlo do zbunjivanja ili nepotrebnog gubljenja vremena, ali iz rada sa studentima je utvrđeno da je takav način rada uglavnom štetan u edukativnom procesu. Stoga je zaključeno da je potrebno što češće uključivati najvažnije pojmove i principe u izvođenje vežbe kako bi studenti što lakše mogli da zaključie koji su pojmovi i principi ključni za razumevanje gradiva. U praksi ovo je postignuto čestim ponavljanjem određenih principa i zadataka prilikom izvođenja laboratorijske vežbe kako bi se kod studenata stvorila rutina vezana za ključne momente u gradivu.

Na prethodnoj slici je prikazano da je za simulaciju korišćen programski paket MATLAB (Matrix Laboratory), dok je kod hardverske implementacije korišćen Code Composer Studio. Bitno je napomenuti da korisnik nije svestan postojanja ovih programa. Oni se ovde koriste u funkciji unutrašnjih modula i kao takvi se mogu zameniti drugim modulima. Npr. u slučaju korišćenja hardvera drugog proizvođača bilo bi potrebno promeniti međukompajler, a što se tiče simulacije mogu se koristiti drugi matematički programi kao što je npr. Octave.

Matlab predstavlja crnu kutiju koja prima ulazne numeričke podatke i daje izlazne numeričke podatke prema nekoj funkciji. Praktično je moguće napisati grafičko okruženje u Matlabu koje bi prikazivalo rezultate obrade digitalnih signala, ali bi to značilo uvođenje dva velika problema:

- Takav grafički sistem bi se morao projektovati i realizovati, a zatim bi se studenti morali prvo obučavati kako da ga koriste pre nego što bi mogli da pristupe radu na laboratorijskim vežbama što bi oduzimalo puno vremena koje bi se moglo iskoristiti za izlaganje gradiva
- Takav sistem ne bi rešio povezivanje sa hardverom, pa bi se dodatno vreme moralo trošiti na upoznavanje dodatnih programskih paketa koji nemaju nikakvih zajedničkih tačaka sa Matlabom (npr. Code Composer Studio).

Iz izloženog se vidi potreba za programom koji bi se bazirao u što većoj meri na savladavanju bitnih principa iz gradiva, a ne na upoznavanje programskih interfejsa raznih softverskih paketa.

Po pravilu, elementi koji čine ovakav sistem (softver) mogu se podeliti na dve grupe: radni okvir (framework) i njegovi dodaci. Dakle, u osnovi je framework koji se može proširivati jedinicama koje mogu da naprave korisnici u skladu sa svojim ciljevima i potrebama. Jedan od glavnih zadataka framework-a je da obezbedi nezavisnost softvera i hardvera, tj. da obezbedi podršku za više hardverskih platformi bez potrebe da korisnik

menja framework ili način programiranja. Ovo u stvari znači da razvijeni programski paket pristupa implementaciji algoritma sa apstraktnog nivoa, dok fizičku implementaciju opisanog algoritma vrši kompajler za datu hardversku platformu. Arhitektura framework-a je po pravilu objektno orijentisana i implementirana je u ANSI C programskom jeziku. Osnovna funkcionalnost framework-a se može proširiti pomoću softverskih dodataka. Dodatke možemo uklopiti sa osnovom samo u obliku kompatibilnih komponenti. Na kraju, funkcionalnost celog sistema zavisi od primenjenih softverskih jedinica (modula). Dakle, framework sadrži samo osnovne module koji potpomažu algoritme, tj. omogućuju razvoj dopunskih jedinica koje se mogu dalje podeliti u dve velike grupe: (1) jedinice za obradu podataka i (2) jedinice za vršenje upravljačkih funkcija koje mogu promeniti redosled obrade podataka.

Cilj interaktivnog razvojnog sistema u realnom vremenu je provera toka razvoja i rada elemenata koji sačinjavaju sistem na ciljnom procesoru i mogućnost upoređivanja rezultata sa simulacionom rezultatima. Tako dobijena metoda se može primeniti u interaktivnom razvoju obezbeđivši postepenu proveru i integraciju.

Razvojni sistem je dizajniran za razvoj aplikacija za digitalnu obradu signala, mada se može slobodno proširivati i koristiti na drugim poljima isto tako efektivno, kao npr. za pisanje tehničke dokumentacije gde se može upotrebiti UML struktura razvijena u programskom paketu. Aplikacije za digitalnu obradu signala su procesorski veoma zahtevne i tačnosti proračuna su od izuzetno velikog značaja. Kao primer možemo uzeti FIR (Finite Impulse Response) filter. Dizajniranje i vrednovanje filtra možemo izvršiti u MATLAB programskom paketu. Na osnovu dobijenog rezultata možemo doći do zaključka da projektovani filter vrši željeni zadatak. Sa razvojnim sistemom se proverava već tokom samog razvoja da li se i na ciljnom procesoru filter ponaša kao u simulaciji. Korišćenjem razvojnog sistema se ispituje da li se poklapaju karakteristike projektovanog filtra u MATLAB programskom paketu i karakteristike filtra na ciljnom procesoru. Dobijeni rezultati se razlikuju pošto ciljni procesor drugačije skladišti koeficijente i drugačije vrši aritmetičke operacije. Ovaj problem se lako može otkriti i korigovati korišćenjem razvojnog sistema već u ranoj fazi razvoja.

Modelovanje koje se koristi u razvojnom sistemu je veoma efektivno za razvoj i testiranje algoritama, iz razloga što se projektovani UML dijagram lako implementira u softver kao pipeline za obradu gde se koraci poklapaju sa koracima koji su naznačeni u UML dijagramu. Modifikacije izvršene na UML dijagramu trenutno se ogledaju na

pipeline-u za obradu. Prednosti ove metode su razvoj i testiranje algoritama na visokom nivou i dobijanje rezultata iz realnog okruženja.

Metoda interaktivnog razvoja u realnom vremenu u digitalnoj obradi ima brojne prednosti kod razvoja aplikacija. Ubrzava se razvojni tok baš iz razloga što je u ranoj fazi moguće testiranje algoritma filtra, kao i korekcija eventualnih nedostataka. Integracija u već postojeći razvojni alat (MATLAB) olakšava primenu ove razvojne metode.

REZULTATI / RESULTS

Kao mera za ocenjivanje uticaja na unapređenje savlađivanja izložene materije koristi se prosek ocena iz školskih 2006/2007 i 2007/2008 godina, kada nije izvođen eksperiment, kao i prosek iz školskih 2008/2009 i 2009/2010 godina u kojima je izvođen eksperiment.

Pošto broj studenata na Visokoj tehničkoj školi strukovnih studija u Subotici retko prelazi cifru od 30 studenata po smeru, odlučeno je da se studenti ne dele na dve testne grupe, već da se programski paket primeni na celu generaciju studenata u toj školskoj godini. Analiza se zatim vršila u odnosu na prosečnu ocenu studenata iz prethodne generacije. Kao dodatna “kontrolna grupa” korišćene su prosečne ocene istih studenata iz bliskog predmeta na kojem se nije menjao koncept nastave primenom ovako prilagođenih softverskih alata i metoda, što je dato kao dodatna referenca u tabeli 2 gde su predstavljene ocene studenata u istom periodu iz predmeta Digitalna obrada signala. Upoređenjem te dve tabele jasno se vidi da nije došlo do sličnih unapređenja proseka studenata iz predmeta Digitalna obrada signala. Ovime se potvrđuje da je napredak studenata iz predmeta Digitalna obrada zvuka i slike rezultat unapređenja izvršenih u edukacionom procesu.

Tabela 1 Povećanje proseka ocena studenata koji su u eksperimentima koristili razvijeni sistem na predmetu Digitalna obrada zvuka i slike

Školska godina	Prosek	Eksperiment
2006/2007	7,08	ne
2007/2008	7,05	ne
2008/2009	7,58	da
2009/2010	7,74	da

Tabela 2 Referentni prosek iz predmeta Digitalna obrada signala

Školska godina	Prosek
2006/2007	6,63
2007/2008	6,89
2008/2009	6,71
2009/2010	7,02

Analizom rezultata iz Tabele 1 jasno se vidi da je došlo do značajnijeg povećanja proseka ocena studenata u godinama kada je korišćen softverski paket. Uspešnost primenjenog softverskog paketa je evidentna zato što su svi upisani studenti tih školskih godina pristali da učestvuju u eksperimentu. Studenti ranijih godina, pre eksperimenta su u nastavi koristili programski paket MATLAB i programski paket Code Composer Studio.

Kao dodatni podatak u prilog uspešne primene razvijenog obrazovnog softvera može da posluži podatak da na drugim predmetima nije zabeležen ovakav napredak u prosečnoj oceni istih studenata.

Eksperiment je izveden u laboratorijama Visoke tehničke škole strukovnih studija u Subotici u trajanju od dve godine. Eksperiment je izvođen sa studentima treće godine elektro i informatičkog smera iz predmeta Obrada zvuka i slike. Proučavanjem literature vezane za edukacione alate primećeno je da ispitivanja i potvrda rezultata nisu rađeni u saradnji sa studentima i nisu se bazirala na nekim konkretnim merenjima. Pošto usvajanje gradiva od strane studenata u velikoj meri zavisi od njihove motivisanosti i inspirisanosti metodama predavanja odlučeno je da se sprovede anketa među studentima koji učestvuju u sprovođenju eksperimenta radi utvrđivanja uticaja eksperimenta na studente i njihovo interesovanje za predmet.

Pošto nije došlo do izmene nastavnog plana i programa iz posmatranog predmeta, a sa obzirom da su grupe studenata male prosek ocena iz školskih godina pre izvođenja eksperimenta se uzima kao kontrolna grupa.

Važno je napomenuti da učešće studenata nije bilo obavezno kao i da učešće u eksperimentu nije donosilo nikakve dodatne bodove na ispitu iz predmeta. Razvijeni softverski paket se nije koristio u ispitivanju studenata.

Studenti su bili anketirani pre i posle korišćenja softverskog paketa. Svaki student je popunio anketu čiji rezultati su prikazani u Tabeli 3. Podaci predstavljeni u tabeli su prikupljeni u periodu od dve godine, 2008 i 2009. Ukupan broj studenata koji su učestvovali u testiranju je 149. Analizom odgovora studenata utvrđeno je da je softverski paket

pozitivno uticao na unapređenje znanja studenata. Ovaj vizuelni i interaktivni alat nije samo povećao brzinu usvajanja znanja kod studenata nego je takođe imao veliki uticaj na podizanje interesovanja studenata na za datu oblast.

Analizom ankete utvrđen je pozitivan uticaj na motivisanost studenata za materiju iz predmeta Obrada zvuka i slike. Važno je napomenuti da anketa ne predstavlja praktično merilo uspešnosti korišćenog edukacionog alata već daje uvid u prihvaćenost razvijene metode u redovima studenata.

Tabela 3 Rezultati ankete

Pitanje	Veoma dobro	Dobro	Delimično	Malo
Oceni svoje poznavanje digitalnih filtara pre pohađanja ovih vežbi!	2	9	59	79
Da li si razumeo matematičke modele filtara pre pohađanja ovih vežbi?	8	43	69	29
Da li te je alat motivisao za rad na vežbama?	38	51	37	23
Koliko je tvoje iskustvo vezano za projektovanje digitalnih filtara pre pohađanja vežbi?	51	35	52	11
Da li su se eksperimenti na vežbama nadovezali na tvoje prethodno znanje iz oblasti digitalnih filtara?	15	80	39	15
Da li je alat lak za korišćenje sa intuitivnim interfejsom?	68	55	24	2
Koliko je, po tvom mišljenju, ovaj alat doprineo tvom savladavanju materije?	80	48	14	7
Kako ocenjuješ svoj rad sa ovim alatom u pogledu lakoće korišćenja i unapređenja tvog znanja?	94	36	14	5
Da li su eksperimenti na vežbama predstavljali demonstraciju principa utvrđenih na teorijskoj nastavi?	62	59	24	4

Postavka hardvera: na slici 2 je predstavljen hardver korišćen u laboratoriji.

Kutija sa desne strane je signal generator koji je preko crvene žice povezan sa ulaznim portom DSP razvojnog okruženja. U kutiji u sredini se nalazi DSK razvojna ploča. Izlazni port iz DSK ploče je preko plave žice povezan sa zvučnicima.

Slika 2 Postavka hardvera



DISKUSIJA / DISCUSSION

Postoje razna istraživanja u vezi sa realizacijom razvojnog okruženja za digitalnu obradu niskofrekventnih signala koja omogućavaju promenu parametara tokom rada. Nedostaci ovakvih rešenja su složena implementacija u već postojeća rešenja, tj. u većini slučajeva se radi o zasebnim aplikacijama koje dokazuju samo jednu zasebnu tvrdnju i nisu predviđena da se povezuju sa drugim programskim paketima radi postizanja boljih rezultata ili proširenja oblasti primene.

Oblasno specifični (Domain Specific) jezici se sve češće primenjuju, ali u polju digitalne obrade signala nije raširena njihova upotreba. Ukazuje se potreba za jednim takvim modernim oblasno specifičnim jezikom, koji bi generalno mogao da se koristi na polju DSP razvoja i koji bi nezavisno od oblasti olakšavao rad istraživača.

Iako su DSL jezici sve zastupljeniji u mnogim oblastima, u digitalnoj obradi signala još uvek nisu postali standard. Samim tim u najvećem broju slučajeva ulaže se energija u definisanje i razvoj sintakse i semantike jezika. Zbog toga je većina razvijenih jezika još uvek bazirana na tekstualnom zadavanju komandi koje se uglavnom specificiraju za rešavanje pojedinačnih zadataka u obradi signala [1], [2]. Ovo čini rešavanje jednostavnih zadataka prilično lakim, ali onemogućava konstruisanje kompleksnih sistema na lak i

intuitivan način. Postojeći jezici su tekstualno orijentisani i uglavnom bazirani na objektno orijentisanim metodama. Glavna preokupacija se u većini slučajeva svodi na rešavanje specifičnih problema, a ne na definisanje osnovnih gradivnih blokova koji lako mogu da se kombinuju u složenije algoritme [3], [4]. Najveći broj razvijenih programskih paketa koristi simulacije za proveru svojih rezultata dok je analiza na hardveru u realnom vremenu zapostavljena [5], [6]. Analizom postojećeg stanja u ovoj oblasti došlo se do zaključka da postoji prostor za razvoj DSL jezika na bazi grafičkih gradivnih elemenata čija bi namena bila što lakše i intuitivnije spajanje u kompleksnije algoritme sa mogućnošću testiranja performansi na realnim platformama.

Nakon proučavanja dostupne literature [7] izabran je UML kao pogodna opcija za modelovanje iz sledećih razloga:

- UML je prepoznat kao standard i postao je veoma zastupljen u industriji
- UML poseduje mehanizme za proširenje (stereotipovi, obeležene vrednosti, napredne provere opsega itd) što znači da je dozvoljeno projektovanje dodatne semantike koja se lako može integrisati u postojeći UML
- UML se modeluje preko mehanizma metamodela što dozvoljava da se definiše metamodel koji može da nadogradi postojeći UML u slučaju potrebe
- Korišćenjem objektno orijentisanog pristupa u UML-u definisanje klase se vrši preko interfejsa klase i ponašanja (behaviour) klase. Ovakvo razdvajanje između definicije i instance klase dozvoljava razvoj biblioteka sa ponovno upotrebljivim (re-usable) komponentama. Dodatna prednost objektno orijentisanog pristupa je sposobnost da se definiše nova komponenta preko nasleđivanja osobina iz druge komponente što opet unapređuje mogućnost ponovnog korišćenja (re-use) postojećih komponenti.

Nadalje, UML ne zavisi od neke posebne metodologije pa se može definisati sopstvena metodologija koja najviše odgovara rešavanju postavljenog problema. Drugi projekti u oblasti sistema u realnom vremenu ili ugrađenih (embedded) sistema su takođe izabrali UML kao jezik za modelovanje. Npr. HASOC metodologija [8] proširuje UML-RT kako bi se uključila anotacija sa mapiranim informacijama. U tom radu autori su predložili povezivanje kapsula sa dodatnim osobinama kao što su sinhroni protok podataka i kodirane mašine stanja. Druga istraživačka grupa je predložila UML profil za ugrađene sisteme [9], koji olakšava modelovanje hardverske platforme i kvantifikovanje QoS (Quality of Service) performansi.

ZAKLJUČAK / CONCLUSION

Zbog svoje interaktivne i grafičke prirode ovo softversko okruženje je dobar kandidat za aplikacije u edukativnim kursevima iz oblasti digitalne obrade signala. Pokazalo se da je pre korišćenja ovog alata studentima dovoljno dva časa teorijskih predavanja i dva časa uvodne nastave u računarskoj laboratoriji.

Korišćenjem ovog softverskog okruženja studentima se omogućuje efikasno:

- razumevanje osnovnih principa FIR i IIR filtara
- povezivanje parametara filtara sa realnim kolom
- izvođenje praktičnih eksperimenata u laboratoriji
- projektovanje praktičnog filtara prema proračunatim teorijskim modelima
- unapređenje znanja iz oblasti digitalne obrade signala u što kraćem vremenu.

LITERATURA / REFERENCES

[1] J.Xiong, J.Johnson, R.Johnson, D.Padua. SPL: A Language and Compiler for DSP Algorithms. PLDI 2001 ISBN:1-58113-414-2

[2] The Design and Implementation of a Domain-Specific Language for Network Performance Testing, Scott Pakin, IEEE Transactions on Parallel and Distributed Systems, October 2007 (Vol. 18, No. 10) pp. 1436-1449.

[3] E. Axelsson, K. Claessen, G. Dvai, Z. Horvth, K. Keijzer, B. Lyckegrd, A. Persson, M. Sheeran, J. Svenningsson, and A. Vajda. Feldspar: a domain specific language for digital signal processing algorithms. In 8th ACM/IEEE Int. Conf. on Formal Methods and Models for Codesign, 2010

[4] G. Dévai, Z. Gera, Z. Horváth, G. Páli, M. Tejfel, "A domain specific language for DSP", 8th International Conference on Applied Informatics January 29, 2010, Eger, Hungary

[5] Real-Time Digital Signal Processing: Implementations and Applications, Sen M. Kuo, Bob H. Lee, Wenshun Tian, 2006

[6] Oliveira N., Pereira M.J., Henriques P., Cruz D., Visualization of Domain Specific Program's Behavior, 5th IEEE International Workshop on Visualizing Software for Understanding and Analysis (co-located with ICSM 2009), Edmouton, Canada, September 2009

[7] Grant Edmund Martin, Dr. Wolfgang Müller, "UML for SOC design", Springer; Softcover reprint of hardcover 1st ed. 2005 edition, ISBN-13- 978-1441938299

[8] A. Cuccuru, P. Marquet and J.L. Dekeyeser, „UML2 as an ADL Hierarchical Hardware Modeling“, Technical report 5166, Institut National de Recherche en Informatique et en Automatique, April 2004

[9] R. Chen, M. Sgroi, G. Martin, L. Lavagno, A. Sangiovanni-Vincentelli and J. Rabaey „Embedded System Design Languages“, Marseille, France, Sempember 2002