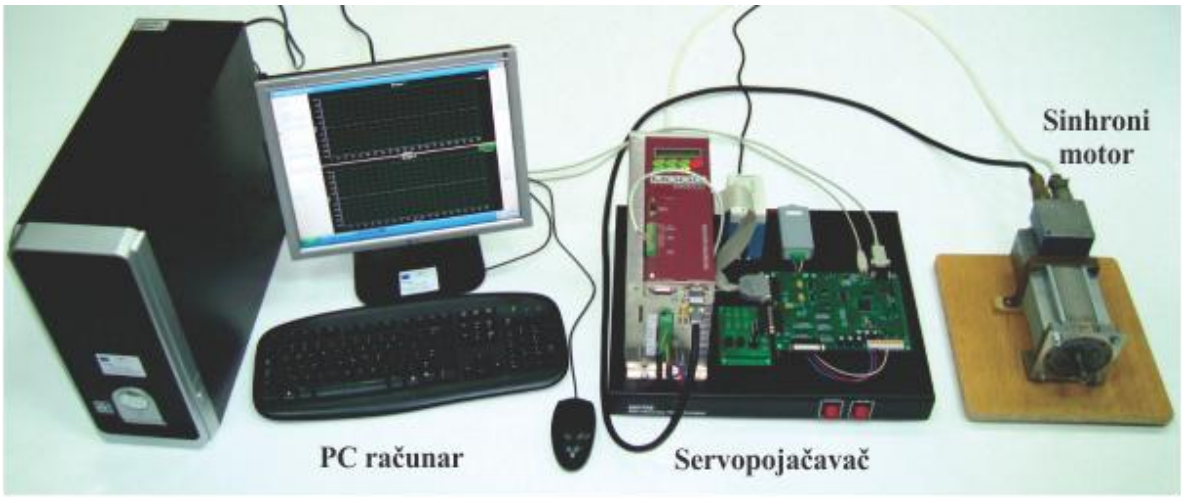
Laboratorija za digitalno upravljanje EPP

OG4EV

OG4DPP

MS1DPP

MS1DP2



Laboratorijske vežbe MOT09

ddc.etf.rs

vozila.etf.rs

masine.etf.rs

ddc@etf.rs

ver - avgust 2011

Autori:

Hardver i softver vežbe je projektovao prof. Slobodan Vukosavić

U izradi postavki je učestvovao magistar Obrad Đorđević

U izradi grafičkog korisničkog interfejsa je učestvovao inž. Milan Vukov

U izradi preliminarnog uputstva su učestvovali Obrad Đorđević i Milan Vukov

U pisanju prvog dela teksta ovog uputstva, do strane 50, učestvovali su magistri Dragan Mihić i Mladen Terzić. U pisanju dela teksta ovog uputstva od strane 50 do strane 100 je učestvovao magistar Nikola Popov. Dodatak A i B su napisali magistri Nikola Popov, Mladen Terzić i Dragan Mihić.

Sadržaj

Uvod

Šta je laboratorijska postavka MOT09

Laboratorijska vežba MOT09 predstavlja digitalno upravljani elektomotorni pogon sa asinhronim motorom i sinhronim motorom sa stalnim magnetima. Motori se napajaju iz trofaznog tranzistorskog invertora sa IGBT tranzistorima. Opremljeni su davačima pozicije na vratilu (rezolver, enkoder). Upravljanje je implementirano na programskom jeziku C i izvršava se na platformi sa digitalnim signalnim procesorom TMS320LF2407. Vežba je opremljena programskim alatima koji omogućuju studentima da na personalnom računaru u realnom vremenu posmatraju oblike napona, struja, momenta, brzine, položaja, fluksa i drugih veličina relevantnih za digitalni regulator struje, vektorsko upravljanje, DTC, kao i za regulisanje brzine i položaja. Tri postavke MOT09 se nalaze u laboratoriji 40A.

Kome je namenjena laboratorijska postavka MOT09

Vežba je namenjena studentima Elektrotehničkog fakulteta u Beogradu koji prate predmete OG4EV, OG4DPP i MS1DPP2 (programi ovih predmeta su dati na ddc.etf.rs, vozila.etf.rs, masine.etf.rs). Pretpostavlja se da student poseduje osnovno znanje iz programskog jezika *C*. Pored vežbanja na 3. i 4. godini studija, kao i vežbanja u okviru master studija, vežba predstavlja univerzalnu hardversku platformu za izradu čitavog niza diplomskih i master radova, projekata i semestralnih radova počevši od istraživanja u oblasti upravljanja generatorima u alternativnim izvorima, pa do upravljanja servo-motorima u sistemima za upravljanje kretanjem.

Koja je korist od vežbanja na MOT09

Kroz praktično izvođenje vežbe korisnik se upoznaje sa osnovnim komponentama digitalno upravljanog elektromotornog pogona, hardverskim resursima za implementaciju digitalnih zakona upravljanja zasnovanih na savremenoj DSP tehnologiji (***DSP*** – ***D****igital* ***S****ignal* ***P****rocessor*) i aspektima kodiranja algoritama za upravljanje u realnom vremenu. Laboratorijska vežba MOT09 omogućava da se student upozna sa osnovnim aspektima softvera za upravljanje u realnom vremenu, strukturama prekida, kao i upravljačkim algoritmima i kodom koji je napisan u programskom jeziku C za digitalne signalne procesore (DSP). Studentu se pruža mogućnost da izradom laboratorijske vežbe shvati i nauči kako se izvode osnovne računske operacije na DSP-u u celobrojnoj aritmetici, kako da koristi osnovne periferije savremenih DSP-ova, kao što su A/D konvertor, interfejs za inkrementalni enkoder, digitalni ulazi i izlazi, PWM izlazi za upravljanje energetskim tranzistorima i osnovne komunikacione veze kao što su asinhrona i sinhrona serijska veza – SCI (RS232) i SPI. Osim toga student se upoznaje sa osnovnim aspektima hardvera, prekidačkim stepenima za napajanje pomoćnih uređaja, drajverskim kolima za upravljanje IGBT tranzistorima, trofaznim tranzistorskim invertorom i motorima sa davačima na vratilu. U pogledu upravljanja student se upoznaje sa implementacijom impulsno širinske modulacije (PWM), sistemom za merenje struje, strujnom regulacijom trofaznih motora naizmenične struje, sa implementacijom vektorskog upravljanja i implementacijom regulacije fluksa i momenta kao i sa regulacijom brzine i pozicije. Studentu se pruža mogućnost da se upozna sa funkcionalnim sistemom kao i da u prvom licu prođe kroz neke od osnovnih vežbi počevši od očitavanja digitalnih ulaza, uključenja digitalnih izlaza – LED (***L****ight* ***E****miting* ***D****iode*), očitavanja analognih signala senzora na vratilu, pa sve do uspostavljanja povratnih sprega i upravljanja brzinom asinhronog i sinhronog motora.

Postavka sa AM i postavka sa sinhronim motorom

Postoje dve varijante laboratorijske vežbe, prva u kojoj se upravlja asinhronim motorom sa kaveznim rotorom i druga u kojoj se upravlja sinhronim servo-motorom sa stalnim magnetima na rotoru . U okviru prve varijante postoje dve postavke (ACIM1, ACIM2), a u okviru druge jedna (PMSM). Detalji i hardverske osobenosti svake postavke objašnjeni su u prilogu B.

Šta sadrži ovaj dokument?

Dokument je podeljen u 2 dela. U prvom delu se se nalaze laboratorijske vežbe u kojima se studenti detaljnije upoznaju sa digitalnim signal procesorima, njihovim periferijama, prekidima i spoljašnjim interfejsima. Upoznavanje sa procesorima se vrši u 3 celine za koje postoje detaljna uputstva i to:

- Uputstvo za vežbu 1, *Demo vežba*
- Uputstvo za vežbu 2, *DSP periferije*
- Uputstvo za vežbu 3, *U/f upravljanje asinhronim motorom*
- Dodatak A, korišćenje USB osciloskopa
- Dodatak B, opis hardvera.

Dodaci A i B se nalaze na kraju dokumenta i studentima su potrebni jer prikazuju način korišćenja opreme koja se koristi kroz sve laboratorijske vežbe.

Drugi deo uputstva prikazuje rad digitalno kontrolisanog pogona i to kroz 4 laboratorijske vežbe. Za svaku od vežbi napisano je detaljno uputstvo i to sledećim tokom:

- Uputstvo za vežbu 4, **Merenje struje korišćenjem davača struje**
- Uputstvo za vežbu 5, **Podešavanje parametara regulatora struje**
- Uputstvo za vežbu 6, **Podešavanje parametra T_r IFOC strukture**
- Uputstvo za vežbu 7, **Ispitivanje rada IFOC**
- Dodaci 1, 2, 3, 4 koji se odnose na vektorsko upravljanje.

Program vežbanja na MOT09

Planirana su sledeća vežbanja:

a) Demo vežba (povezivanje postavke, U-f upravljanje, vektorsko upravljanje

b) DSP periferije (digitalni ulazi i izlazi, ADC, prekidi, PWM, enkoder, kodiranje)

c) U/f upravljanje asinhronim motorom

d) Merenje struje korišćenjem davača struje (u drugom delu uputstva)

e) Podešavanje parametara regulatora struje (u drugom delu uputstva)

f) Podešavanje parametra T_r IFOC strukture (u drugom delu uputstva)

g) Ispitivanje rada IFOC (u drugom delu uputstva)

Uvodna DEMO vežba

Student se osposobljava za samostalno povezivanje postavke odnosno njenih glavnih hardverskih delova. Zatim se upoznaje sa osnovnim softverom koji se koristi u okviru vežbe kao i sa procedurom programiranja DSP-a (upisivanja koda u procesor). Student će se takođe upoznati i sa načinom povezivanja PC računara sa DSP-om kao i sa grafičkim prozorom (*eng*. GUI – Graphic User Interface) u računaru koji se koristi za prikazivanje analognih i digitalnih signala koje prima računar putem USB veze. Pomenuti GUI će se u okviru vežbi, zbog njegove funkcije i sličnosti sa stvarnim osciloskopom, zvati *USB osciloskop*. Detaljno uputstvo za korišćenje USB osciloskopa dato je u prilogu A.

Sama vežba predstavlja implementaciju U/f i IFOC upravljanja asinhronim motorom ali će se u okviru ove uvodne vežbe zadržati samo na manifestacijama upravljanja to jest, na zadavanje i praćenje brzine obrtanja motora, praćenje struja i napona motora itd. Detalji algoritma upravljanja i odgovarajućeg programskog koda biće detaljnije obrađeni u narednim delovima vežbe.

DSP periferije

Ovaj deo vežbe služi za upoznavanje studenta sa pojmom interapt rutine kao i sa osnovnim periferijama savremenog DSP-a, a to su: ulazno/izlazni portovi, A/D i D/A konvertor, PWM modul i enkoderski interfejs.

U/f regulacija asinhronog motora i merenje struje

Ovaj deo vežbe služi za upoznavanje studenta sa algoritmom U/f upravljanja i njegovim kodiranjem u programskom jeziku C, zatim sa spuštanjem koda u procesor i praćenjem rada pogona uz zadavanje nekih karakterističnih veličina upravljanja. Posebnu celinu u okviru ovog dela vežbe predstavlja upoznavanje studenta sa načinom merenja struje i napona kao i sa osnovnim karakteristikama invertora kojim se upravlja.

Drugi deo uputstva, vežbe 4, 5, 6, 7

Ove vežbe sadrže elemente vežbanja koji su od velike važnosti za razumevanje digitalne kontrole pogona. Na početku se analizira sistem merenja struje, zatim se podešava regulator struje koji se u poslednje dve vežbe koristi u cilju ispitivanja performansi indirektne vektorske kontrole.

Postupak studenata koji žele da rade vežbu

Koliko traje svaka od 7 vežbi

Vreme potrebno za izradu svake celine u okviru laboratorijske vežbe iznosi oko tri sata.

Šta raditi pre vežbe

Od studenata se zahteva da pre dolaska na vežbe detaljno pročitaju i prouče uputstvo i upoznaju se sa osnovnim aspektima vežbe. Laboratorijska vežba se održava u laboratoriji 40A. Pre pristupa vežbama asistenti postavljaju pitanja kako bi proverili osnovnu pripremu studenta za vežbu. Sve informacije vezane za laboratorijsku vežbu student može dobiti od predmetnih asistenata putem e-maila ili u kabinetu 27.

1. Demo vežba

1.1 Ciljevi demo vežbe

- Osposobaljavanje studenta za samostalno povezivanje laboratorijske postavke odnosno njenih osnovnih delova. Detaljniji opis hardverskih delova postavki dat je u prilogu B.
- Uključivanje postavke tj. dovođenje napajanja na dva osnovna dela: energetski i elektronski.
- Upoznavanje sa procedurom kompajliranja i upisivanja programa u DSP.
- Uvežbavanje korišćenja USB osciloskopa za prikaz merenih veličina pogona na ekrenu PC-a. Uputstvo za korišćenje dato je u prilogu A.
- Manifestacije *U/f* i indirektnog vektorskog upravljanja (IFOC) i njihovo kvalitativno poređenje prema odzivu brzine.
- Očekuje se da student sve navedene stavke prođe samostalno prateći detaljno uputstvo.
- **U koliko se desi nešto nepredviđeno neophodno je pozvati dežurnog asistenta!**

1.2 Tok vežbe

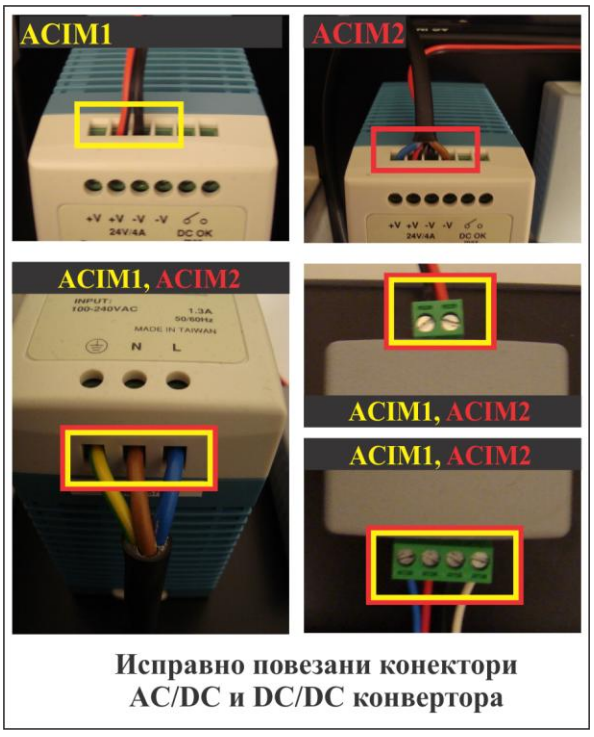
Povezivanje postavke

- Po dolasku u laboratoriju opredeliti se za postavku na kojoj student želi da radi vežbu.
- Potrebno je prvo proveriti da li su kablovi ispravno povezani prema slikama koje slede (stavke A1-A3)
- Zatim student treba da poveže preostale kablove takođe prema priloženom uputstvu (stavke B1-B3)

Proveru povezanosti svih konektora na servopojačavaču treba izvršiti kroz sledeće korake:

A1. Provera povezanosti konektora AC/DC i DC/DC konvertora

Ispravno povezani konektori AS/DC i DC/DC konvertora prikazani su na fotografijama na Slici 1.1.



Slika 1.1 – Fotografije ispravno povezanih konektora AS/DC i DC/DC konvertora

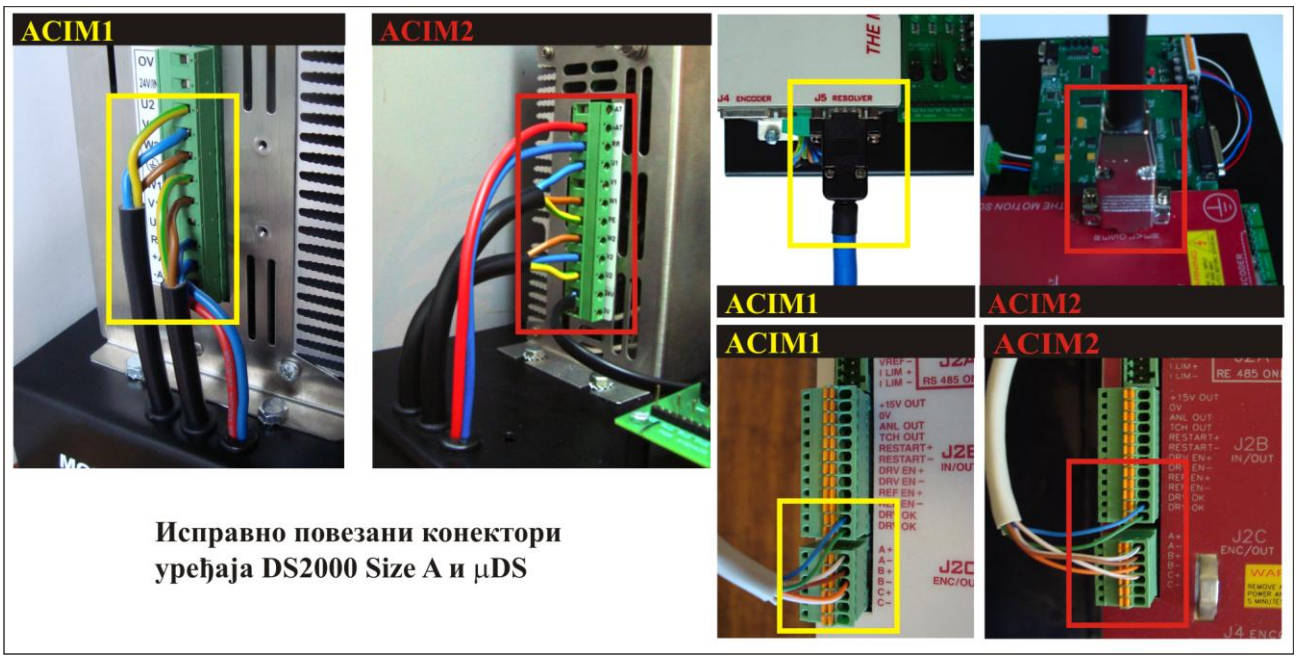
Slika 1.2 – Fotografije ispravno povezanih konektora na JMU-L ploči

A2. Provera povezanosti konektora na JMU-L ploči

Na Slici 1.2 date su fotografije ispravno povezanih konektora na JMU-L ploči.

A3. Provera povezanosti konektora invertora DS2000

Na Slici 1.3 prikazane su fotografije ispravno povezanih konektora invertora DS2000 Size A i µDS.



Slika 1.3 – Fotografije ispravno povezanih konektora invertora DS2000 Size A i µDS

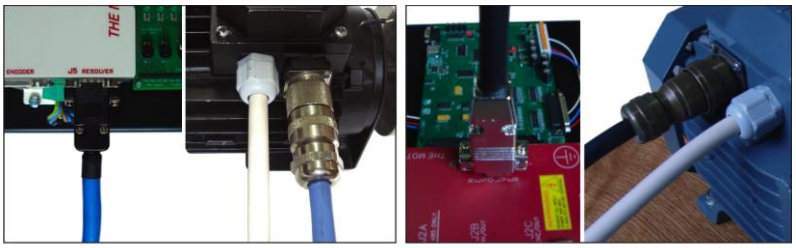
Na dalje treba pratiti sledeće korake za povezivanje glavnih hardverskih delova tj. servopojačavača sa PC računarom i motorom:

B1. Povezivanje servopojačavača sa motorom. Servopojačavač se povezuje sa motorom pomoću 2 kabla: napojnog i rezolverskog.

Napojni kabl motora (kabl bele boje koji izlazi iz motora, odnosno jednoznačno se povezuje na odgovarajući konektor motora) treba povezati sa odgovarajućim konektorom MOTOR sa zadnje strane servopojačavača. Žuto zelenu žicu koja viri iz napojnog kabla motora treba zaviti pod „leptir" šraf označen sa GND, koji se takođe nalazi sa zadnje strane servopojačavača. Prikaz ispravno povezanog napojnog kabla sa servopojačavačem, prikazan je na Slici 1.4.

Slika 1.4. Prikaz ispravno povezanog napojnog kabla na servopojačavač

Rezolverski kabl motora treba povezati na odgovarajući konektor na samom motoru, dok drugi kraj treba povezati na port J5 Resolver invertora u koliko prethodno nije povezan. Prikaz ispravno povezanog rezolverskog kabla na strani invertora, i na strani motora, za obe laboratorijske postavke prikazan je na Slici 1.5.

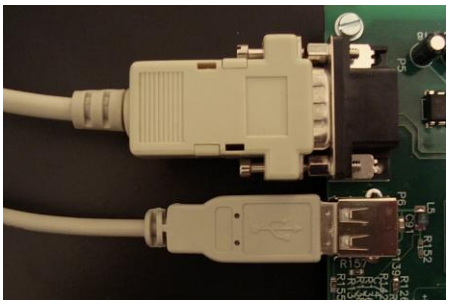


а) MOT09 ACIM1 б) MOT09 ACIM2

Slika 1.5. Prikaz ispravno povezanog rezolverskog kabla, za obe postavke (a, b) na strani servopojačavača i na strani motora

B2. Povezivanje servopojačavača sa PC računarom.

Servopojačavač se povezuje sa PC računarom pomoću 2 kabla: USB i serijskog kabla. Računar i servopojačavač povezivati isključivo kada su i računar i servopojačavač isključeni. Povezati serijski kabl između serijskog porta PC računara i serijskog porta JMU-L ploče, kao i USB kabl između USB porta PC računara i USB porta JMU-L ploče. Na Slici 1.6 je dat prikaz ispravno povezanih prethodno pomenutih kablova sa JMU-L pločom.



Slika 1.6 - Prikaz ispravno povezanih kablova za komunikaciju JMU-L ploče sa PC računarom

B3. Dovođenje napajanja.

Kada povežemo servopojačavač i računar, treba povezati napojni kabl u odgovarajući konektor AC IN koji se nalazi pri sredini sa zadnje strane servopojačavača. Prikaz zadnje strane servopojačavača sa ispravno povezanim kablovima dat je na Slici 1.7.

Slika 1.7 - Prikaz ispravno povezanog kabla za napajanje (AC IN)

Ovim je završen proces povezivanja postavke i sada se može preći na proceduru njenog puštanja u rad i demonstracije U/f, a zatim i vektorskog upravljanja.

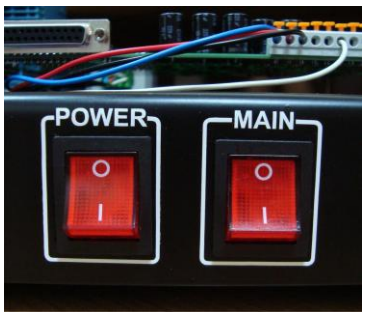
Demonstracija U/f upravljanja

CILJ:

- U okviru ovog dela vežbe student će se prvo upoznati sa puštanjem postavke u rad.
- Zatim sledi upoznavanje sa procedurom kompajliranja i upisivanja programa u DSP.
- Nakon toga sledi demonstracija U/f upravljanja

1. Puštanje postavke u rad

Nakon ispravnog povezivanja svih hardverskih komponenti laboratorijske vežbe može se pristupiti uključenju postavke u rad i upisivanju programa u DSP. Postavka se uključuje tako što se PC računar uključi na standardni način, a prekidači *Main* i *Power* postave u položaj ON (proizvoljnim redosledom). Na Slici 1.8 prikazani su navedeni prekidači u OFF položaju (pre uključivanja).



Slika 1.8 - Prekidači Power i Main u OFF položaju (pre uključenja)

Može se reći generalno da se MAIN prekidač koristi za uključivanje elektronskog (digitalnog) dela, DSP JMU-L ploče, dok se prekidač POWER koristi za uključivanje energetskog dela, odnosno invertora.

Nakon ovog koraka sledi demonstracija U/f upravljanja.

2. Ponalaženje potrebnih fajlova za U/f upravljanje

Za svaku laboratorijsku postavku se koristi softver namenjen isključivo toj postavci. Softver se nalazi u odgovarajućem projektnom folderu za demo vežbu. Korišćenje neodgovarajućeg softvera može dovesti do nekontrolisanog ponašanja motora i njegovog oštećenja!

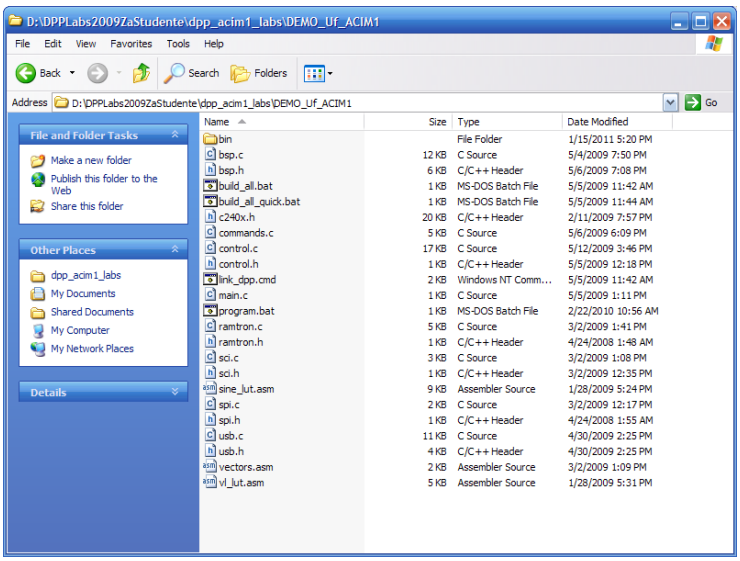
U tabeli 1.1 se nalaze podaci o projektnim folderima za obe laboratorijske postavke sa opisom fajlova koji se koriste u okviru demo vežbe, a koji sadrže odgovarajući algoritam U/f upravljanja. Ovi fajlovi su različiti za svaku od postavki.

Lab. Postavka	**ACIM1**	**ACIM2**
Projektni direktorijum	*DEMO_Uf_ACIM1*	*DEMO_Uf_ACIM2*
Fajlovi od značaja	Opis	
main.c	***Glavni projektni fajl***	
control.c	***Implementacija kontrolnih algoritama***	
build_all.bat	***Skripta za kompajliranje, linkovanje i pravljenje izvršnog fajla.***	
program.bat	***Skripta za upisivanje (engl. download) programa u DSP***	

Tabela 1.1. Projektni fajlovi upravljačkih softvera u laboratorijskim postavkama od posebnog značaja.

Folder ***DEMO_Uf_ACIM1*** je potrebno pronaći na sledećoj putanji: *D:\DPPLabs2009ZaStudente\ dpp_acim1_labs\ DEMO_Uf_ACIM1*

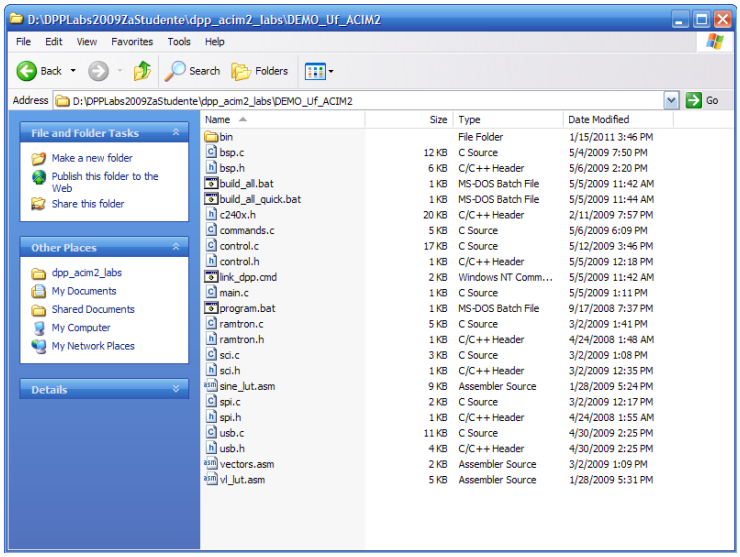
Sadržaj ovog foldera dat je na Slici 1.9 gde se mogu uočiti i fajlovi koji su navedeni u Tabeli 1.1.



Slika 1.9 - Sadržaj foldera ***DEMO_Uf_ACIM1***

Folder ***DEMO_Uf_ACIM2*** je potrebno pronaći na sledećoj putanji: *D:\DPPLabs2009ZaStudente\ dpp_acim2_labs\ DEMO_Uf_ACIM2*

Sadržaj ovog foldera dat je na Slici 1.10. gde se mogu uočiti i fajlovi koji su navedeni u tabeli 1.1.

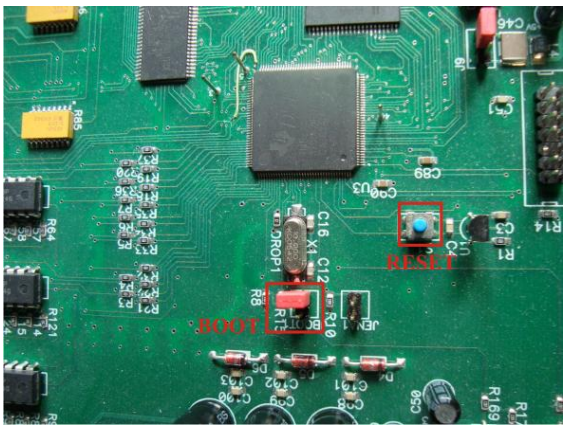


Slika 1.10 - Sadržaj foldera ***DEMO_Uf_ACIM2***

U zavisnosti od izabrane postavke otvoriti ogovarajući projektni folder.

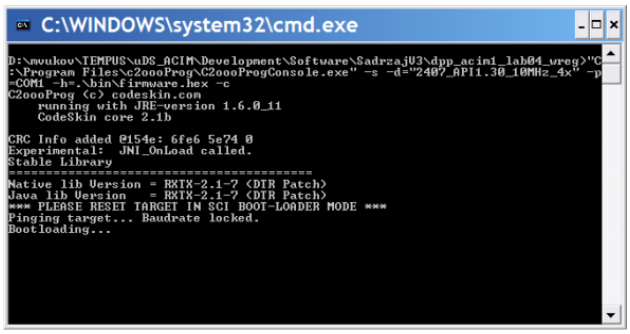
3. Procedura upisivanja koda u DSP

1. U odgovarajućem projektnom folderu pokrenuti skriptu ***build_all.bat***, koja služi za kompajliranje i povezivanje („linkovanje“) projektnih fajlova. Potrebno je ispratiti sekvencu poruka koje se ispisuju na ekranu i nakon svakog pauziranja na ekranu pritisnuti bilo koji taster na tastaturi. Na kraju, kada se prozor zatvori, u tekućem folderu će se nalaziti folder ***bin*** i u njemu fajl ***firmware.hex***, tj. fajl koji treba upisati u DSP. Ukoliko se pojave bilo kakvi problemi u ovom koraku, pozvati dežurnog asistenta/laboranta.
2. Uočiti džamper **BOOT** i **RESET** dugme na digitalnoj ploči JMU-L, Slika 1.11. Da bi mogao da se upiše kod u DSP potrebno je da se kratkospoji džamper **BOOT** (tj. džamper treba postaviti iz položaja na Slici 1.11 u položaj u kom pokriva oba pina). Zatim je potrebno pritisnuti dugme **RESET**. Sada je DSP spreman za upis programa.

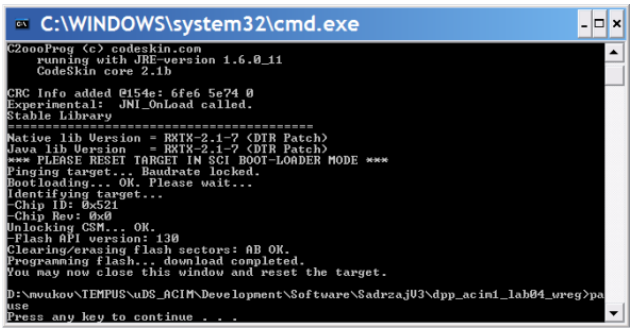


Slika 1.11 - JMU-L ploča – džamper BOOT i dugme RESET

3. Pokrenuti skriptu ***program.bat***. Ukoliko je sve u redu, na ekranu računara će se otvoriti *command prompt* prozor i njegov sadržaj će biti sličan kao na Slici 1.12 **(Nakon završrtka pritisnuti bilo koji taster na tastaturi za izlazak)**. Procedura upisivanja koda traje 10-20 sekundi. Po završetku upisivanja programa u DSP, taj prozor bi otprilike trebao da izgleda kao na slici Slici 1.13. Da bi se krenulo sa izvršavanjem upisanog programa, potrebno je skinuti džamper (i staviti ga na jedan pin kao na Slici 1.11). Nakon toga treba resetovati DSP pritiskom na dugme **RESET**. Ukoliko se javi neka greška ili nepredviđena situacija izaći iz prozora pritiskom na bilo koje dugme. Proveriti da li su sve veze u redu. Ukoliko jesu, pozvati dežurnog asistenta/laboranta u pomoć.



Slika 1.12. Upisivanje programa u DSP – uspešno uspostavljanje veze



Slika 1.13. Upisivanje programa u DSP – uspešan završetak upisivanja programa

4. Prikaz relevantnih veličina pomoću USB osciloskopa

Da bi se pokrenuo USB osciloskop i prikazale željene veličine potrebno je proći kroz tačke 1-6 iz priloga A.

NAPOMENE:

A) Spisak kanala i njima odgovarajućih veličina koje se koriste u ovoj vežbi je dat u tabeli 1.2.

Kanal	Veličina	Vrednosti	Digitalni ekvivalenti na ekranu računara	
			ACIM1	ACIM2
0, 1, 2	i_a , i_b , i_c	1 [A]	200	80
		5 [A]	1000	400
		10 [A]	2000	800
3, 4, 5	u_a , u_b , u_c	±300 [V]	±32768	
6	ω_r	±1500 [rpm]	Prikazuje se u [rpm]	
7	f_r	±200 [Hz]	±32768	

Tabela 1.1. Veličina koje se prikazuju na graficima i njihovo skaliranje

B) U okviru tačke 6 priloga A treba čekirati kanale iz tabele 1.2 da bi se omogućio prikaz potrebnih veličina.

5. Zadavanje referntne vrednosti frekvencije

Vrši se tako što se u polju pored labele *Mnemonic* unese *FR*, a u polje pored labele *Data* broj *20*. Pritiskom na dugme *Write* pogonu se zadaje referentna frekvencija f_r = 20 [Hz] tj. u DSP se ovom komandom upisuje vrednost željene frekvencije napona napajanja motora. Nakon ovoga motor bi trebao da počne da se obrće. Brzina se ne menja naglo jer se referentna učestanost menja po rampi do zadate vrednosti. Ovo je neophodno da bi se izbegli nagli skokovi struje motora pri promeni frekvencije. Koristeći se stavkama 7 i 8 iz priloga A snimiti rampu po kojoj se uspostavlja brzina. Sa komandom *capture screen* snimiti ekran na kome se vidi rampa brzine i to kopirati u jedan Word fajl. Na Slici 1.15 je dat primer rampe referentne frekvencije.

U narednom koraku pokazaćemo kao se može menjati nagib ove rampe tj. brzina uspostavljanja zadate frekvencije.

NAPOMENA: Motor će se obrtati nešto manjom brzinom od brzine kojoj odgovara zadata frekvencija zbog brzine klizanja u praznom hodu.

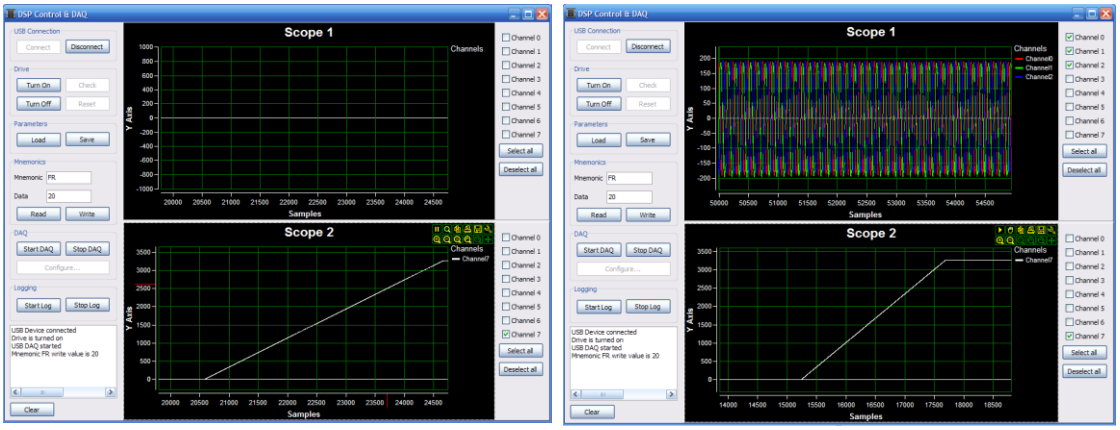
6. Promena brzine uspostavljanja referentne frekvencije (nagiba rampe)

Prvo je potrebno otvoriti program ***control.c*** u kome je implementiran algoritam U/f upravljanja. On se nalazi u odgovarajućem projektnom folderu kao što je navedeno u tabeli 1.1. Sada je potrebno uočiti deo koda kojim se zadaje komanda za frekvenciju. Taj deo koda se nalazi od 476 do 496 linije.

```
473 // Opis:
474 //  - Funkcija koja implementira U/f regulaciju
475 // ****************************************************************************
476 void Drive1_UF( void )
477 {
478   // Priprema nove frekvencije za U/f, referenca "ide po rampi"
479   if ( drive1_state == ON )
480   {
481     // drive1_uf_mscounter x 100us period expired
482     if ( ++drive1_uf_mscounter == 10 )
483     {
484       if ( drive1_f_ref > drive1_f )
485       { drive1_f++; }
486       else if ( drive1_f_ref < drive1_f )
487       { drive1_f--; }
488
489       drive1_uf_mscounter = 0;
490     }
491   }
492   else
493   {
494     drive1_angle.full = 0;
495     drive1_f = 0;
496   }
497
498   drive1_angle.full += drive1_f * drive1_param_scale_angle;
```

Slika 1.14 – Deo koda kojim se zadaje komanda za frekvenciju

U kodu na Slici 1.14 treba uočiti sledeće promenljive: ***drive1_f, drive1_f_ref*** i ***drive1_uf***_*mscounter*. Promenljiva ***drive1_f*** predstavlja komandu frekvencije koja se zadaje invetoru. Promenljiva ***drive1_f_ref*** predstavlja referentnu vrednost frekvencije koja je zadata u koraku 11 (ona ima vrednost promenljive FR). Promenljiva drive1_uf_mscounter predstavlja broj izvršenih instrukcijskih ciklusa DSP-a. Jedan ciklus traje 100μs. Kada navedena promenljiva dostigne zadatu vrednost koja je u ovom slučaju 10 (tj. nakon 1ms), ispunjen je uslov u *if* naredbi (uokvirena crvenom bojom na Slici 1.14). Tada se izvršava set instrukcija koji je uokviren plavom bojom i koji vrši inkrementiranje i dekrementiranje promenljive ***drive1_f*** u koliko je ona manja od promenljive ***drive1_f_ref***.



a) *b)*

Slika 1.15 - Rampa po kojoj se uspostavlja komanda za frekvenciju za korak od: a) 5 ciklusa; b) 3 ciklusa

Ovaj proces inkrementiranja i dekrementiranja promenljive ***drive1_f*** vrši se sve dok ona ne dostigne referentnu vrednost frekvencije tj. vrednost promenljive ***drive1_f_ref***. Može se zaključiti da brzina inkrementiranja naredbe za frekvenciju, a samim tim i brzina dostizanja referentne frekvencije zavisi od zadatog broja ciklusa koji se poredi sa promenljivom ***drive1_uf_mscounter*** (u ovom slučaju 10). Promenom ovog broja menja se nagib rampe po kojoj se uspostavlja zadata frekvencija ***drive1_f_ref***.

Sada je potrebno promeniti pomenuti broj ciklusa na 3. Zatim sačuvati izmene u fajlu ***control.c*** ,zatvoriti ga i ponoviti proceduru upisivanja koda u DSP iz tačke 3. Zatim ponovo zadati referencu frekvencije kao u tački 5. Koristeći se stavkama 7 i 8 iz priloga A snimiti rampu po kojoj se uspostavlja brzina u ovom slučaju. Sada opet koristeći komandu *capture screen* snimiti ekran na kome se vidi rampa brzine i kopirati ga u isti Word fajl iz tačke 5. Na Slici 1.15 a) i b) dat je primer rampe frekvencije za korak od 5 odnosno 3 ciklusa u odgovarajućoj razmeri koja je data u tabeli 1.2. Takođe, posmatrati ubrzanje motora do referentne brzine u ovom slučaju. Nakon toga uporediti rampu brzine u jednom i u drugom slučaju.

Ovim je završena demonstracija U/f upravljanja i sada sledi demonstracija vektorskog upravljanja.

Demonstracija indirektnog vektorskog upravljanja

CILJ:

- Ponavljanje procedure puštanja postavke u rad, kompajliranja i upisivanja odgovarajućeg programa za ovu vežbu u DSP
- Demonstracija indirektnog vektorskog upravljanja

Za razliku od U/f upravljanja koje predstavlja upravljanje u otvorenoj sprezi, algoritam indirektnog vektorskog upravljanja (IFOC - *eng.* ***I****ndirect* ***F****ield* ***V****ector* ***C****ontrol*) podrazumeva postojanje strujnog i brzinskog regulatora. Vektorskim upravljanjem asinhroni motor se pretvara u aktuator momenta sa veoma brzim odzivom na odgovarajuću komandu momenta odnosno komandu brzine u koliko postoji brzinska petlja. Cilj ovog dela vežbe je da se posmatra odziv motora na zadatu komandu brzine i poređenje ubrzanja motora u ovom slučaju sa slučajem U/f upravljanja.

1. Puštanje postavke u rad

Ponoviti proceduru puštanja postavke u rad koja je opisana u tački 1 poglavlja **Demonstracija U/f upravljanja**.

2. Ponalaženje potrebnih fajlova za indirektno vektorsko upravljanje

Softever za indirektno vektorsko upravljanje nalazi se u odgovarajućem projektnom folderu za demo vežbu.

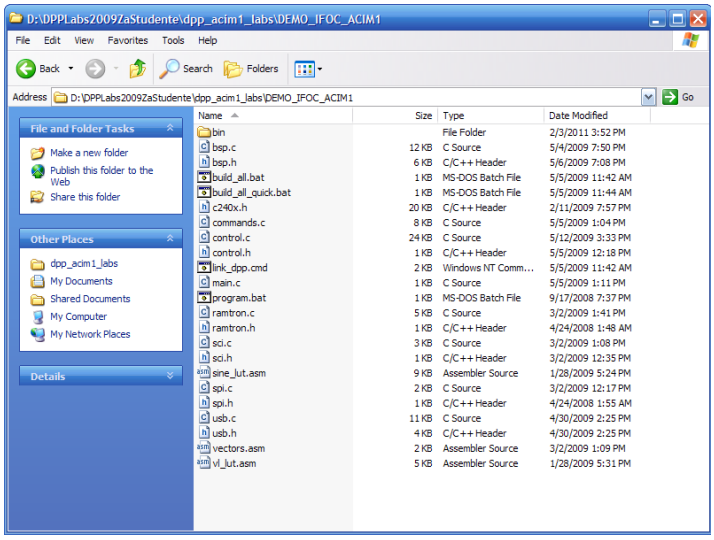
U sledećoj tabeli se nalaze podaci o projektnim folderima vezanim za IFOC. Ovi fajlovi su različiti za svaku od postavki.

Lab. Postavka	**ACIM1**	**ACIM2**
Proj. Folder	***DEMO_IFOC_ACIM1***	***DEMO_IFOC_ACIM2***
Fajlovi od značaja	Opis	
main.c	***Glavni projektni fajl***	
control.c	***Implementacija kontrolnih algoritama***	
build_all.bat	***Skripta za kompajliranje, linkovanje i pravljenje izvršnog fajla.***	
program.bat	***Skripta za upisivanje (engl. download) programa u DSP***	

Tabela 1.3. Projektni fajlovi upravljačkih softvera u odgovarajućim laboratorijskim postavkama.

Folder ***DEMO_IFOC_ACIM1*** je potrebno pronaći na sledećoj putanji: *D:\DPPLabs2009ZaStudente\ dpp_acim1_labs\ DEMO_IFOC_ACIM1*

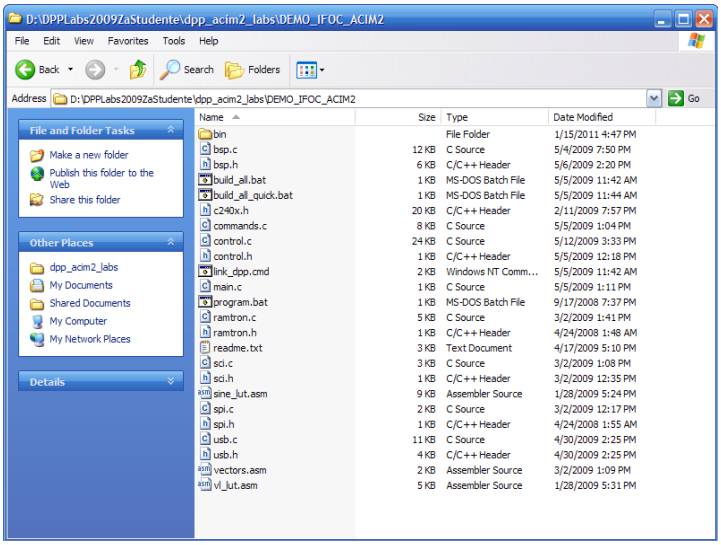
Sadržaj ovog foldera dat je na Slici 1.16 gde se mogu uočiti i fajlovi koji su navedeni u tabeli 1.3.



Slika 1.16 - Sadržaj foldera ***DEMO_IFOC_ACIM1***

Folder ***IFOC_demo_ACIM2*** je potrebno pronaći na sledećoj putanji: *D:\DPPLabs2009ZaStudente\ dpp_acim2_labs\ DEMO_IFOC_ACIM2*

Sadržaj ovog foldera dat je na Slici 1.17 gde se mogu uočiti i fajlovi koji su navedeni u tabeli 1.3.



Slika 1.17 - Sadržaj foldera ***DEMO_IFOC_ACIM2***

3. Procedura upisivanja koda u DSP

Ponoviti proceduru upisivanja koda u DSP koja je detaljano opisana u tački 3 poglavlja **Demonstracija U/f upravljanja**.

4. Prikaz relevantnih veličina pomoću USB osciloskopa

Da bi se pokrenuo USB osciloskop i prikazale željene veličine potrebno je proći kroz tačke 1-6 iz priloga A.

NAPOMENE:

A) Spisak kanala i njima odgovarajućih veličina koje se koriste u ovom delu vežbe kao i njihovo skaliranje dato je u tabeli 1.4.

Kanal	Veličina	Vrednosti	Digitalni ekvivalenti na ekranu računara		
			ACIM1	ACIM2	PMSM
0, 1, 2, 3	i_q^*, i_q, i_d^*, i_d	1 [A]	200	80	44
		5 [A]	1000	400	220
		10 [A]	2000	800	445
4	ω_r^*	±1500 [rpm]	Prikazuju se u [rpm]		
5	ω_r				
6	u_a	±300 [V]	±32768		
7	u_b				

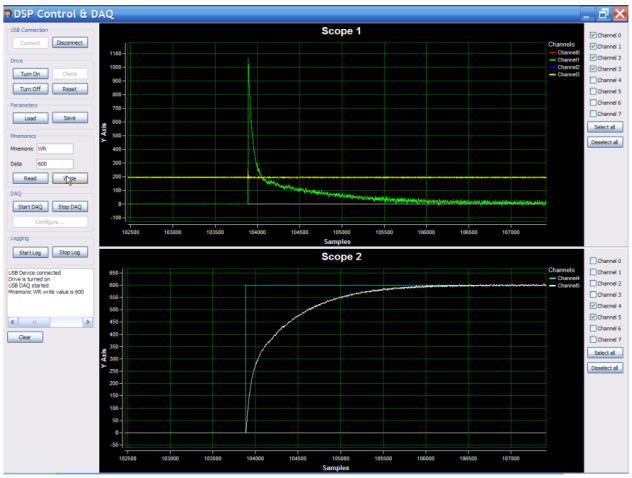
Tabela 1.4. Skaliranje veličina koje se prikazuju na graficima

B) U okviru tačke 6 priloga A treba čekirati kanale iz tabele 1.4 da bi se omogućio prikaz potrebnih veličina i to na prvom grafiku čekirati kanale 0, 1, 2 i 3, a na na drugom kanale 4 i 5.

5. Zadavanje referntne brzine

Vrši se tako što se u polju pored labele *Mnemonic* unese ***WR*** , a u polje pored labele *Data* broj ***1000***. Pritiskom na dugme *Write* pogonu se zadaje referentna brzina $n_r = 1000[o/\min]$ tj. u DSP se ovom komandom upisuje vrednost željene brzine obrtanja motora. Nakon ovoga motor bi trebao da počne da se obrće. Brzina se menja gotovo trenutno. Koristeći se stavkama 7 i 8 iz priloga A „uhvatiti“ i prikazati odziv brzine motora. Sa komandom *capture screen* snimiti ekran na kome se vidi odziv brzine i to

kopirati u jedan Word fajl koji zatim treba sačuvati u odgovarajući projektni folder. Na Slici 1.18 dat je primer odziva brzine motora(donji grafik) i izgled struje motora pri njegovom startu.



Sl 1.18 - Primer trenutka puštanja u rad asinhronog motora

Potrebno je na kraju uočiti razliku u brzini uspostavljanja zadate brzine obrtanja motora u slučaju IFOC i U/f regulacije.

Ovim je završena demonstracija IFOC upravljanja.

2. Periferije DSP-a

2.1 Uvod

Grupa vežbi koje slede (V1 do V5) imaju za cilj upoznavanje studenta sa osnovnim periferijama DSP-a, načinom njihovog funkcionisanja i upravljanja.

V1 upoznaje studenta sa ulaznim i izlaznim portovima DSP-a, tačnije sa programiranjem i upisivanjem S koda u DSP kojim se manifestuje rad ovih periferija. Biće ukazano na osnovne funkcije kojima se postiže upravljanje stanjima pinova.

V2 daje studentu osnovne informacije o pojmu prekida i prekidne rutine i upoznaje ga sa načinom generisanja prekida u DSP-u.

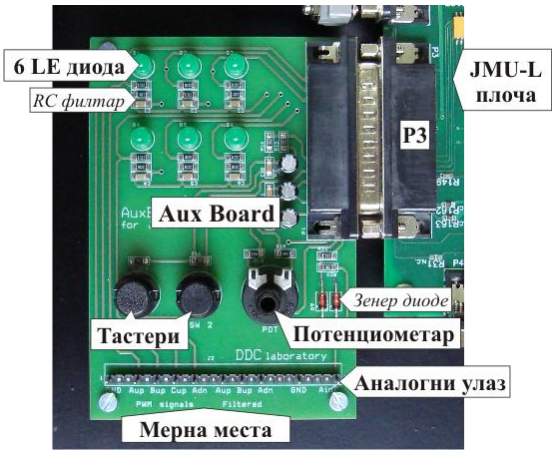
V3 upoznaje studnta sa periferijom DSP-a koja se koristi za konverziju analognih signala u digitalni ekvivalent (ADC modul). Student će saznati neke osnovne stvari o načinu konverzije, iplementaciji odgovarajućeg S koda u procesor i nekim osnovnim algoritmima koji omogućavaju upravljanje ADC modulom.

V4 upoznaje studenta sa modulom za generisanje PWM impulsa, njegovom ulogom, načinom rada, implementacijom S koda koji omogućava njegov rad i osnovnim funkcijama kojim se postiže upravljanje ovim modulom kroz praktičnu demonstaciju rada na nekom od postojećih postavki.

V5 upoznaje studenta sa modulom DSP-a koji obrađuje kvadraturne enkoderske impulse i daje apsolutnu vrednost pozicije motora. U vežbi će se izvršiti upisivanje S koda u procesor i prikaz obrađenih enkoderskih impulsa tj. pozicije na USB osciloskopu.

2.2 Oprema koja se koristi u vežbama

1. Postavke ACIM1, ACIM2 i PMSM. Opis njihovih hardverskih delova je dat u prilogu B.
2. AuxBoard pločica koja predstavlja deo samih postavki. AuxBoard pločica sa označenim osnovnim delovima data je na Slici 2.1. Detaljniji opis ove pločice i njenih delova nalazi se u prilogu B.



Slika 2.1 - AuxBoard pločica sa osnosvnim delovima

3. Digitalni osciloskop *Tectronix* sa naponskom sondom sondom.

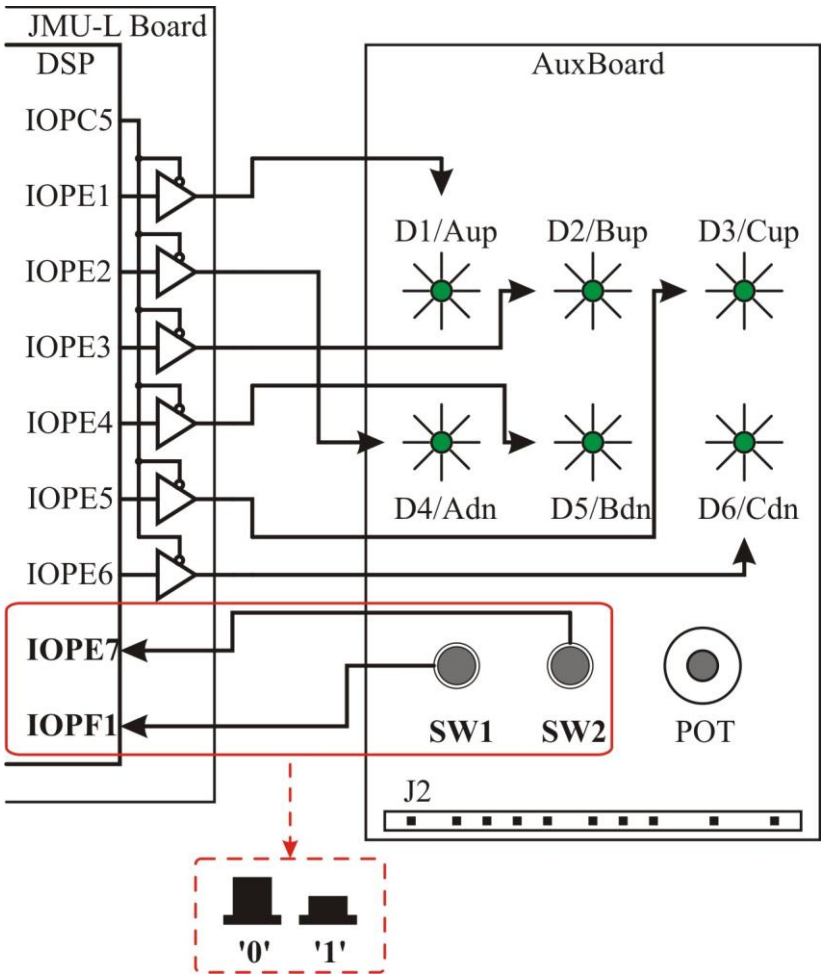
2.3 VEŽBA 1 - Ulazni i izlazni portovi

Ciljevi vežbe

Cilj prve vežbe je upoznavanje studenta sa konfigurisanjem i kontrolom stanja na pinovima dva od ukupno šest portova DSP-a (PORT A, PORT B,…, PORT F). Svaki port sadrži šest pinova, a za svaki pin se

određenom instrukcijom može odrediti da li je ulazni ili izlazni. Manifestacija upravljanja pinovima biće prikazana na diodama AuxBoard pločice. Student će u okviru vežbi proći kroz sledeće korake:

- U DSP se upisuje pripremljeni S kod u okviru koga se određenim setom instrukcija pinovima E1 do E6 dodeljuje se funkcija izlaza. Ti pinovi su fizički povezani sa LE diodama na AuxBoard pločici (Slici 2.2.). Promena stanja na tim pinovima uključuje odnosno isključuje diode na pločici pri čemu je potrebno naglasiti da prisustvo logičke 0 na pinu uključuje, a prisustvo logičke 1 isključuje diodu.



Slika 2.2 – Prikaz povezanosti LED i tastera sa odgovarajućim pinovima DSP-a

- U tabeli 2.1 je prikazan redosled bitova 0-7 u registru PEDATDIR koji definišu stanja na pinovima porta E i diode koje su na njih povezane:

PEDATDIR	IOPE7	IOPE6	IOPE5	IOPE4	IOPE3	IOPE2	IOPE1	IOPE0
bitovi registra	X	0 ili 1	0 ili 1	0 ili 1	0 ili 1	0 ili 1	0 ili 1	x
Diode	SW2	D6	D3	D5	D2	D4	D1	-

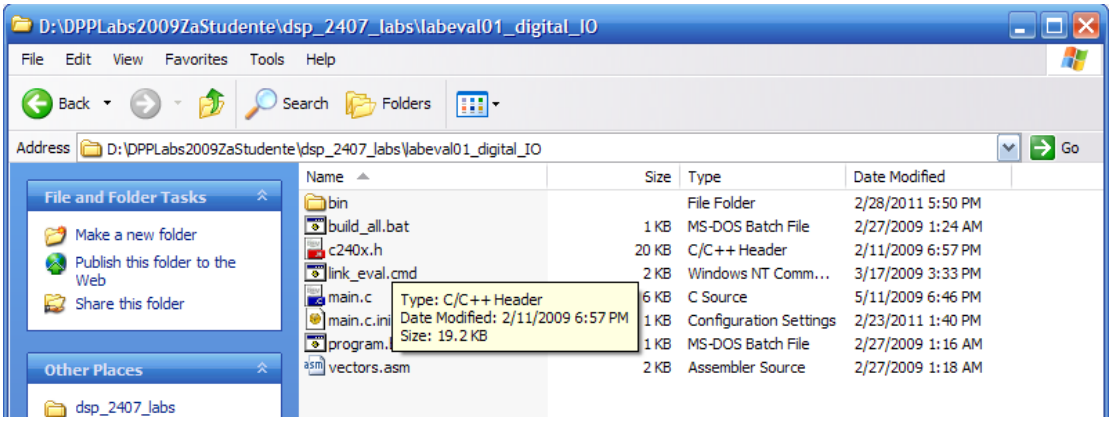
Tabela 2.1 – Bitovi registra koji definiše stanja na pinovima porta E i njima odgovarajuće diode

- U okviru S programa DSP-a implementirana je određena logika kojom će se menjati stanje na izlaznim pinovima.
- Određenim programskim naredbama kao ulazni pinovi konfigurisani su pinovi E7 i F1 na koje su povezani tasteri SW1 i SW2 sa AuxBoard pločice.
- Nakon toga se u okviru datog programa prati stanje ulaznog pina F1 koje zavisi od stanja tastera SW1 (ON/OFF) koji je na ovaj pin priključen (Slika 2.2). U programu će takođe biti implementirano da se promenom stanja na ovom pinu menja dinamika paljenja i gašenja dioda. (Praćenje stanja tastera SW2

odnosno odgovarajućeg pina sa kojim je on povezan nije implementirano u okviru datog programa već se ostavlja studentima kao moguća vežba)

Tok vežbe

1. Odabrati odgovarajuću postavku za izradu vežbe (ASIM1, ASIM2 i PMSM) i na njemu uočiti AuxBoard pločicu i identifikovati njene osnovne delove date na Slici 2.1.
2. Uključiti PC računar i prekidač *main*.
3. Otvoriti odgovarajući projektni folder za ovu vežbu ***labeval01_digital_IO*** koji se nalazi na putanji *D:\DPPLabs2009ZaStudente\dsp_2407_labs\labeval01_digital_IO*, a čiji je sadržaj prikazan na Slici 2.3.



Slika 2.3 – Sadržaj foldera ***labeval01_digital_IO***

main.c je glavni programski modul koji će kasnije biti analiziran i modifikovan u okviru ove vežbe.

4. Ponoviti proceduru kompajliranja programa i njegovog upisivanja u DSP koja je opisana u koraku 3 poglavlja **Demonstracija U/f upravljanja** u Demo vežbi. Kompajliranjem se program napisan u programskom jeziku S *main.c* prevodi u asemblerski jezik i smešta u fajl *main.asm*, a zatim prevodi u mašinski jezik koji se smešta u fajl *firmware.hex* (niz nula i jedinica u heksadecimalnom zapisu) koji se zatim upisuje u DSP.
5. Po završenom upisivanju programa u DSP pritisnuti dugme *reset* na JMU-L ploči pri čemu će diode početi da se pale i gase po određenoj logici koja je definisana u programu. Pritiskom na taster *reset* na JMU-L ploči program počinje da se izvršava od zaglavlja funkcije ***main(void).***
6. Pritisnuti taster SW1 pri čemu se menja dinamika paljenja dioda.
7. Isključiti prekidač *main.*
8. Dvostrukim klikom otvoriti program ***main.c*** koji se nalazi u projektnom folderu.
9. Pronaći deo koda koji je prikazan na Slici 2.4. Ovim delom koda konfigurišu se portovi E i F koji se koriste u ovoj vežbi i to na sledeći način:

   - Prvo se portovima E i F umesto njihove primarne funkcije dodeljuje funkcija izlazno/ulaznog porta tako što se u registar ***MCRC*** upišu odgovarajući bitovi. Ovo je urađeno linijom koda koja je označena sa (1) na Slici 2.4. Bit 0 znači da će ogovarajući pin imati funkciju ulaza/izlaza, a bit 1 da pin zadržava svoju primarnu funkciju. Oznaka ***x*** znači da se odgovarajući pin u datoj vežbi ne koristi.
   - Nakon ovoga je potrebno definisati da li će odgovarajući pin biti ulazni ili izlazni. To se ostvaruje tako što se u gornjih osam bita (bitovi 8-15) registra ***PEDATDIR*** (za port E) i registra ***PFDATDIR*** (za port F) upiše 0 (za funkciju ulaza) odnosno 1 (za funkciju izlaza). To je ostvareno linijom koda označenom sa (2) odnosno (3) na Slici 2.4. Iz linije (2) se može zaključiti da je pinovima od 8 do 14 porta E dodeljena funkcija izlaza (bitovi 1 na odgovarajućim mestima), a pinu 15 funkcija ulaza (bit 0 na odgovarajućem mestu u registru). Zapisano u heksadecimalnom sistemu vrednost gornjih 8 bita

registra ***PEDATDIR*** je ***7F (bin = 01111111).*** Iz linije (3) se može zaključiti da je pinovima 5 i 6 porta F dodeljena funkcija izlaza (bitovi 1 na odgovarajućim mestima), a svim ostalim pinovima funkcija ulaza (bitovi 0 na odgovarajućim mestima). Zapisano u heksadecimalnom sistemu vrednost gornjih 8 bita registra ***PFDATDIR*** je ***60 (bin = 01100000).***

- Na kraju je potrebno inicijalizovati stanja na pinovima tako što se u donjih 8 bita registara ***PEDATDIR*** i ***PFDATDIR*** upiše 1 ili 0 zavisno od željenog inicijalnog stanja pina. Ovo je takođe realizovano linijama koda koje su označene sa (2) i (3) gde je u donjih osam bita registra E upisan heksadecimalni broj ***7F (bin = 01111111)***, a u donjih osam bita registra F heksadecimalni broj ***00(bin = 00000000).*** U tabeli 1.5 se može videti veza pinova porta E i dioda na AuxBoard ploči. Promena stanja pinova uzrokuje paljenje (bit 0) odnosno gašenje (bit 1) odgovarajućih dioda.

```
   ************************************************************************
*    // Konfiguracija funkcija portova
   ************************************************************************
*

   // bitovi     b7    b6    b5    b4    b3    b2    b1    b0
   // PORT E:    QEP3  PWM12 PWM11 PWM10 PWM9  PWM8  PWM7  CLKOUT
   // MCRC0..7:  0      0     0     0     0     0     0     1      = 0x81
!!!

   // bitovi     b15   b14   b13   b12   b11   b10   b9    b8
   // PORT F:    x     PF6   PF5   PF4   PF3   PF2   PF1   QEP4
   // MCRC8..15: x      x     0     0     0     0     0     0      = 0x01
!!!

   (1)     MCRC = 0x0181;
     //
************************************************************************
     // Konfiguracija digitalnih ulaza i izlaza
   ************************************************************************
*

   // PORT E:    QEP3 PWM12  PWM11 PWM10 PWM9  PWM8  PWM7  CLKOUT
   // DIR:       I    O      O     O     O     O     O     O      = 0x7F
   // DAT INIT:  x    H      H     H     H     H     H     H      = 0x7F

     PEDATDIR = 0x007F;      // DAT
   (2)     PEDATDIR = 0x7F7F;      // DIR

   // PORT F:    x    LEDG   LEDR  RESET DREN  DROK2 DROK1 QEP3
   // DIR:       x    O      O     I     I     I     I     I      = 0x60
   // DAT INIT:  x    L      L     x     x     x     x     x      = 0x00

     PFDATDIR = 0x0000;      // DAT
   (3)     PFDATDIR = 0x6000;      // DIR
```

Slika 2.4 - Deo koda programa **`main.c`** kojim se konfigurišu portovi E i F

10. Sada pronaći deo koda koji je prikazan na Slici 2.5. Ovo je glavni deo koda kojim se implementira logika paljenja i gašenja dioda. U nastavku je dat opis pojedinih delova koda koji je neophodno pročitati da bi se razumela logika upravljanja portovima i omogućio uspešan rad sa ostatkom vežbe.

```
          unsigned long period = 1000000;

          void Wait( void )
(1)       {
                unsigned long counter;

                for ( counter = 0; counter < period; counter++ ) {}
          }

          int main( void )
          {
          unsigned int leds_p1[ 6 ] =
(2)                       { 0x7F03, 0x7F05, 0x7F09, 0x7F11, 0x7F21, 0x7F41 };

                  unsigned int leds_p2[ 6 ] =
                          { 0x7F7D, 0x7F7B, 0x7F77, 0x7F6F, 0x7F5F, 0x7F3F };

                  unsigned int leds_p3[ 6 ] =
                          { 0x7F7D, 0x7F77, 0x7F5F, 0x7F7B, 0x7F6F, 0x7F3F };

                  unsigned int i=0;
                  unsigned int sw1_state = 0, sw1_state_old = 0;
                  unsigned int sw2_state = 0, sw2_state_old = 0;

                  unsigned int perioddiv = 0;

                  Init();

                  PCDATDIR &= CLR5;

          while ( 1 )    //pocetak beskonacne petlje
                {
                      sw1_state = PFDATDIR & 0x0001;
(3)                   if ( ( sw1_state ) && ( sw1_state_old == 0 ) )
(4)                   {
                            perioddiv = ( perioddiv + 1 ) & 0x0003;
(5)
                            period = 1000000 >> perioddiv;
(6)                   }
                      sw1_state_old = sw1_state;
(7)
                      if ( ++i == 6 )
(8)                   { i = 0; }

                      PEDATDIR = leds_p1[ i ];
(9)
                      Wait();
(10)
                }
          }
```

Slika 2.5 - Deo koda programa ***main.c*** kojim se konfigurišu portovi E i F

Opis dela koda sa Slike 2.5:

(1) Funkcija ***Wait***. Sastoji se samo od ***for*** petlje koja u sebi sadrži promenljivu ***counter*** koja se u svakom prolasku kroz petlju inkrementira za 1. Iz petlje se izlazi kada promenljiva ***counter*** postane jednaka promenljivoj ***period*** koja se definiše u glavnom programu. Promenom vrednosti promenljive ***period*** u okviru glavnog programa menja se trajanje izvršavanja ***for*** petlje odnosno funkcije ***Wait***. Time se pri

svakom pozivanju funkcije ***Wait*** generiše pauza u izvršavanju glavnog programa odnosno pauza između postavljanja novog stanja izlaznih pinova (odnosno dioda). Dužina pauze odnosno brzina uspostavljanja novog stanja na diodama određena je vrednošću promenljive ***period*** i vrednošću takta procesora.

(2) Definicija i inicijalizacija promenljivih. Ovde se definišu nizovi ***leds_p1, leds_p2*** i ***leds_p3***. Članovi nizova predstavljaju određena stanja na diodama (0 = upaljena, 1 = ugašena) predstavljena heksadecimalnim brojem. Poslednje dve cifre heksadecimalnog broja sadrže stanja šest dioda na AuxBoard pločici. Svaki niz u stvari definiše određeni redosled paljenja i gašenja dioda i u zavisnosti od željene logike bira se jedan od tri prethodno definisana niza ili se pravi novi po spostvenoj logici. Uzastopnim dodeljivanjem vrednosti članova odabranog niza registru ***PDATDIR*** menjamo stanje na diodama (9).

(3) Provera da li je prvi pin porta F na logičkoj jedinici i dodela rezultata poređenja promenljivoj ***sw1_state*** koja predstavlja stanje prvog tastera SW1.

(4) Ukoliko je sadašnje stanje pina F1 odnosno promenljive ***sw1_state*** na logičkoj jedinici (pritisnut taster SW1), a prethodno stanje je nula, uvećavamo *delilac perioda brojanja* koji je definisan promenljivom ***perioddiv*** (skraćujemo pauzu između postavljanja stanja na diodama smanjenjem promenljive ***period***) (6).

(5) Ova linija koda predstavlja brojač po modulu četiri. U njoj se uvećava delilac perioda brojanja ***perioddiv*** za 1. Kada u jednom trenutku njegova vrednost postane 4 (odnosno ***0x0003*** u heksadecimalnom zapisu) on se resetuje na 0.

(6) Kao što je već rečeno period brojanja u funkciji ***Wait*** određen je promenljivom ***period***. Period od pola sekunde odgovara (približno) brojanju od 0 do 1000000, a šiftovanje ovog broja (1000000) za vrednost ***perioddiv*** (0, 1, 2, 3) odgovaraće vremenskim periodima od 1/2 s, 1/4 s, 1/8 s i 1/16 s. To šiftovanje je upravo definisano ovom linijom koda.

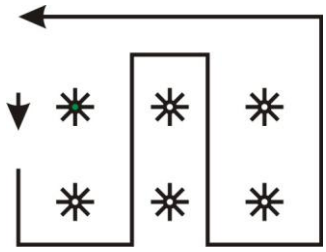
(7) Staro stanje tastera SW1 postaje trenutno stanje.

(8) Brojač po modulu 6. Ovde se određuje indeks niza stanja na izlaznim pinovima kojih ima ukupno 6.

(9) Odgovarajuću diodu palimo tako što u registar ***PEDATDIR*** upisujemo odgovarajuću vrednost iz niza stanja ***leds_p1*** kao što je već objašnjeno u (2).

(10) Funkcija koja generiše pauzu u izvršavanju programa. Period je definisan promenljivom ***period***.

11. Sada formirati niz ***leds_p4*** takav da se diode pale po redosledu koji je prikazan na Slici 2.6. Takođe modifikovati funkciju ***Wait*** tako da se diode pale i gase na svakih 125 ms.



Slika 2.6 - Redosled po kome treba da se pale diode

12. Sačuvati načinjene izmene u programu ***main.c*** i zatvoriti ga.

13. Uključiti prekidač *main*.

14. Ponoviti proceduru kompajliranja i snimanja koda u procesor koja je objašnjena u koraku 3 poglavlja 1.3.

15. Pritiskom na dugme *reset* na JMU-L ploči i tastera SW1 na *AuxBoard* ploči verifikovati načinjene izmene u kodu.

16. Isključiti prekidač *main*.

17. Kraj vežbe!

Zadaci i pitanja

1. Izvršiti izmenu nad originalnim kodom tako da stanja na diodama mogu da se menjaju na svakih 1/4 s, 1/8 s, 1/16 s, 1/32 s. Stanja na diodama treba da se menjaju periodično, kao po šemi ispod.

   oxx -> xox -> xxo

   oxx xox xxo

2. Koja je uloga bita 15-8 u PEDATDIR, a koja je uloga bita 7-0?
3. Koja je uloga kompajlera, drugim rečima, koja je veza između datoteka *ime.c*, *ime.asm* i *firmware.hex?*
4. Na kom mestu počinje izvršavanje programa po pritisku tastera *reset*?
5. Koji efekat na izvršavanje programa ima izostavljanje linije Wait()?
6. Koja je razlika između operatora & i operatora &&?

***NAPOMENA**: Nakon izvršenih modifikacija u zadatku 1 podrazumeva se da student treba da kompajlira i upiše program u DSP, kao i da verifikuje izvršavanje programa (ovo važi za sve vežbe gde se traži bilo kakva modifikacija).*

2.4 VEŽBA 2 - Sistemi prekida

Cilj vežbe

- Kroz ovu vežbu u okviru poglavlja **Teorijske osnove** student će se upoznati sa pojmom prekida i prekidne rutine kao i sa modulima koji generišu prekide.
- U okviru poglavlja **Tok vežbe** student će se kroz praktičnu vežbu upoznati sa postojanjem izvesnog brojača u DSP-u i načinom njegovog brojanja odnosno promene njegovog sadržaja u vremenu.

Teorijske osnove

- Sistem prekida omogućava prekid procesora tokom izvršavanja glavnog programa u cilju obrade događaja višeg prioriteta koji mogu biti inicirani ili od strane hardvera ili od strane softvera. Na ovaj način se izbegava potreba da procesor neprekidno proverava da li se desio neki događaj. Tako se postiže brže vreme odziva i manje zauzeće procesora.
- Događaj višeg prioriteta koji se izvršava kada nastupi prekid naziva se prekidna ili interapt rutina (eng. *interrupt routine*). Na početku svake interapt rutine potrebno je ustanoviti uzrok prekida. Nakon toga sledi kod koji korisnik želi da bude izvršen pri datom interaptu. U okviru interapt rutine potrebno je ponovo dozvoliti interapte (*interrupt enable*) jer procesor automatski briše dozvole za interapt pri izvršenju interapt rutine. Primer interapt rutine dat je sledećim linijama koda:

```
#define INT2_T1PINT     0x0027      // Timer 1 period interrupt

void interrupt INT2_ISR( void )
{
static unsigned int int2_source;

int2_source = PIVR;           // Save interrupt source

...
if ( int2_source == INT2_T1PINT )
{
      DoControlJob();
}
...
INT_ENABLE();                 // Enable interrupts, global
}
```

- U procesoru postoje digitalni brojači (tajmeri) koji broje impulse koji dolaze iz kvarcnog oscilatora procesora. Pošto je frekvencija oscilatora poznata i stalna, brojači se mogu koristiti za merenje vremena pa otuda i naziv tajmer. Vrednost brojača se automatski uvećava/smanjuje određenom unapred definisanom brzinom koju definiše takt procesora kao i određena konstanta kojom se ovaj takt deli, a koju nazivamo preskaler (*prescaler*). Vrednost brojača se upisuje u odgovarajući registar TxCNT (x = 1,2,3,4).
- Brojači broje do određenog broja nakon čega se resetuju ili broje unazad u koliko su bidirekcioni. Na taj način, ako bi se posmatrao sadržaj bidirekcionog brojača, on bi se menjao po nekakvoj testerastoj funkciji vremena. Nagib testere, odnosno brzina brojanja brojača definisana je preskalerom.
- Brojači odnosno tajmeri se mogu podesiti tako da pri dostizanju određene vrednosti izbrojanih impulsa generišu prekid procesoru pri čemu se poziva prekidna rutina.
- 16-bitni registar T1CON koristi se za konfigurisanje rada tajmera1 (brojača1). U njemu se podešavaju sledeći bitovi da bi se obezbedio željeni rad tajmera:

  12-11: izbor načina brojanja – treba upisati redom **0 1** da bi se izabrao režim bidirekcionog brojanja (testera)

  10-8: izbor preskalera – upisati npr. **1 1 1** u koliko se želi preskaler kojim se smanjuje brzina brojanja brojača 128 puta. U koliko je potrebno povećati učestanost interapta treba smanjiti preskaler ili smanjiti vrednost u registru T1PR koji sadrži period brojanja brojača.

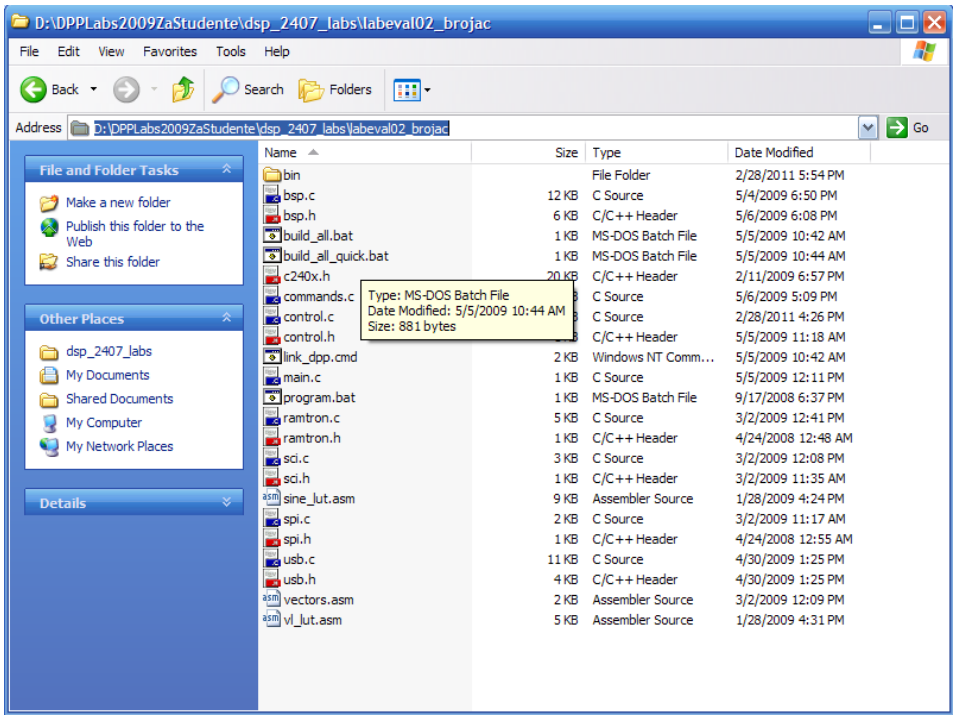
  6: aktiviranje i deaktiviranje tajmera (timer enable/disable): upisati 1 da bi se tajmer aktivirao

  5-4: selektovanje izvora impulsa koji uvećavaju sadržaj brojača (clock source select) - upisati **0 0** za impulse sa internog oscilatora
- U okviru vežbe će na ekranu USB osciloskopa biti prikazan sadržaj jednog bidirekcionog brojača DSP-a. Koristiće se brojač čiji se sadržaj smešta u registar T1CNT.

Tok vežbe

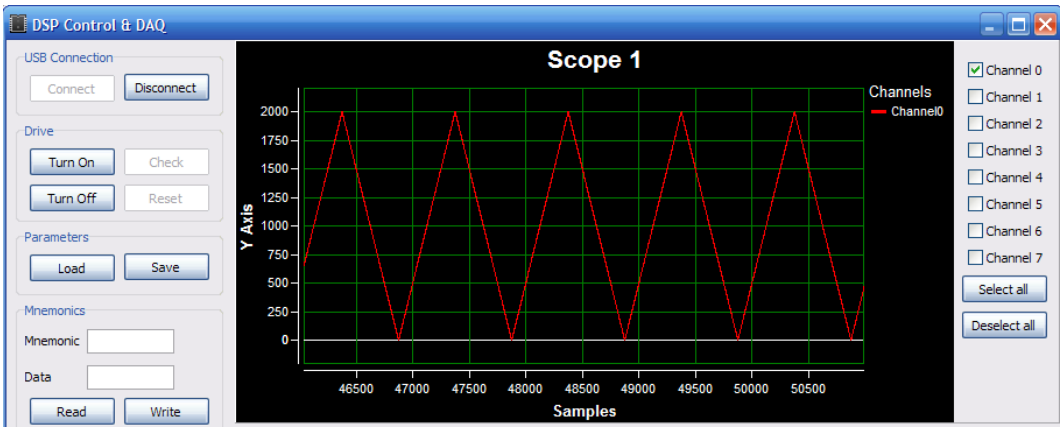
Da bi se posmatrao sadržaj jednog od brojača na USB osciloskopu, potrebno je izvršiti sledeće korake:

1. Otvoriti projektni folder koji se nalazi na sledećoj putanji: *D:\DPPLabs2009ZaStudente\dsp_2407_labs\labeval02_brojac*. Sadržaj ovog foldera prikazan je na Slici 2.7



Slika 2.7 – Sadržaj projektnog foldera labeval02_brojac

2. Ponoviti proceduru kompajliranja programa i njegovog upisivanja u DSP koja je opisana u koraku 3 poglavlja **Demonstracija U/f upravljanja** u Demo vežbi.
3. Nakon toga treba pokrenuti program *dspgui.exe* koji se nalazi u istoimenom folderu na putanji *D:\DPPLabs2009ZaStudente\dsp_2407_labs\dspgui.*
4. Čekirati *Channel* 0 na gornjem grafiku pa zatim pritisnuti na dugme *Connect*.
5. Nakon ovoga na gornjem grafiku treba da se pojavi tresterasta funkcija koja predstavlja sadržaj brojača koji se posmatra i koja izgleda kao na Slici 2.8.



Slika 2.8 – Sadržaj brojača procesora prikazan na ekranu USB osciloskopa

Pošto je ova vežba čisto demonstraciona neće se iznositi detalji vezani za kod programa.

Zadaci i pitanja

1. Koji brojač DSP-a se koristi za generisanje prekida?
2. Koji događaj na ovom brojaču daje prekid?
3. Kako treba inicijalizovati ključne bite GPTCON i TxCON registara da bi se dobio adekvatan rad tajmera?
4. Kako treba izmeniti kod da bi se učestanost interapta prepolovila?
5. Kako treba pisati interapt rutinu u jeziku C?
6. Nabrojati statusne bite koje treba postaviti u registrima za periferije (tajmere), u registrima za interapte kao i u glavnim statusnim registrima da bi interapti mogli raditi? (pogledati poglavlje 6 str. 208-224 (od ukupno 538 strana) *TMS320LF/LC240xA DSP Controllers Reference Guide*, Texas Instruments)
7. Kako treba modifikovati interapt rutinu da bi mogla biti prekinuta interaptom višeg prioriteta?
8. Pogledati asemblersku verziju C fajla () koja ima nastavak ASM). Uočiti *save* i *restore* sekvencu instrukcija. Opisati u par reči o čemu se tu zapravo radi.

2.5 VEŽBA 3 - Analogno/digitalna konverzija (ADC)

Ciljevi vežbe

- Upoznavanje studenta sa osnovnim karakteristikama modula za AD konverziju DSP-a i pokretanjem konverzije.
- Upisivanje u DSP već pripremljenog programa kojim se demonstrira proces AD konverzije analognog naponskog signala koji se dovodi sa potenciometra AuxBoard pločice i prevodi u digitalni ekvivalent. Taj broj će se zatim prikazivati pomoću USB osciloskopa na ekranu računara.
- Upoznavanje studenta sa pojedinim programskim naredbama kojima se upravlja procesom AD konverzije.

U okviru poglavlja **Hardverske veze i tok signala** je opisana veza hardverskih delova AuxBord pločice koji se koriste u okviru ove vežbe. Izrada same vežbe počinje od poglavlja **Tok vežbe**. Nakon toga sledi poglavlje **Teorijske osnove** u kome je ukratko opisan modul za AD konverziju DSP-a kao i sam proces konverzije. U poglavlju **Softver-kratak osvrt na kod** opisane su osnovne funkcije u S kodu pomoću kojih se upravlja procesom AD konverzije. Ova dva poglavlja se preporučuju zainteresovanim studentima radi proširivanja znanja ali nisu neophodna za izradu same vežbe.

Hardverske veze i tok signala

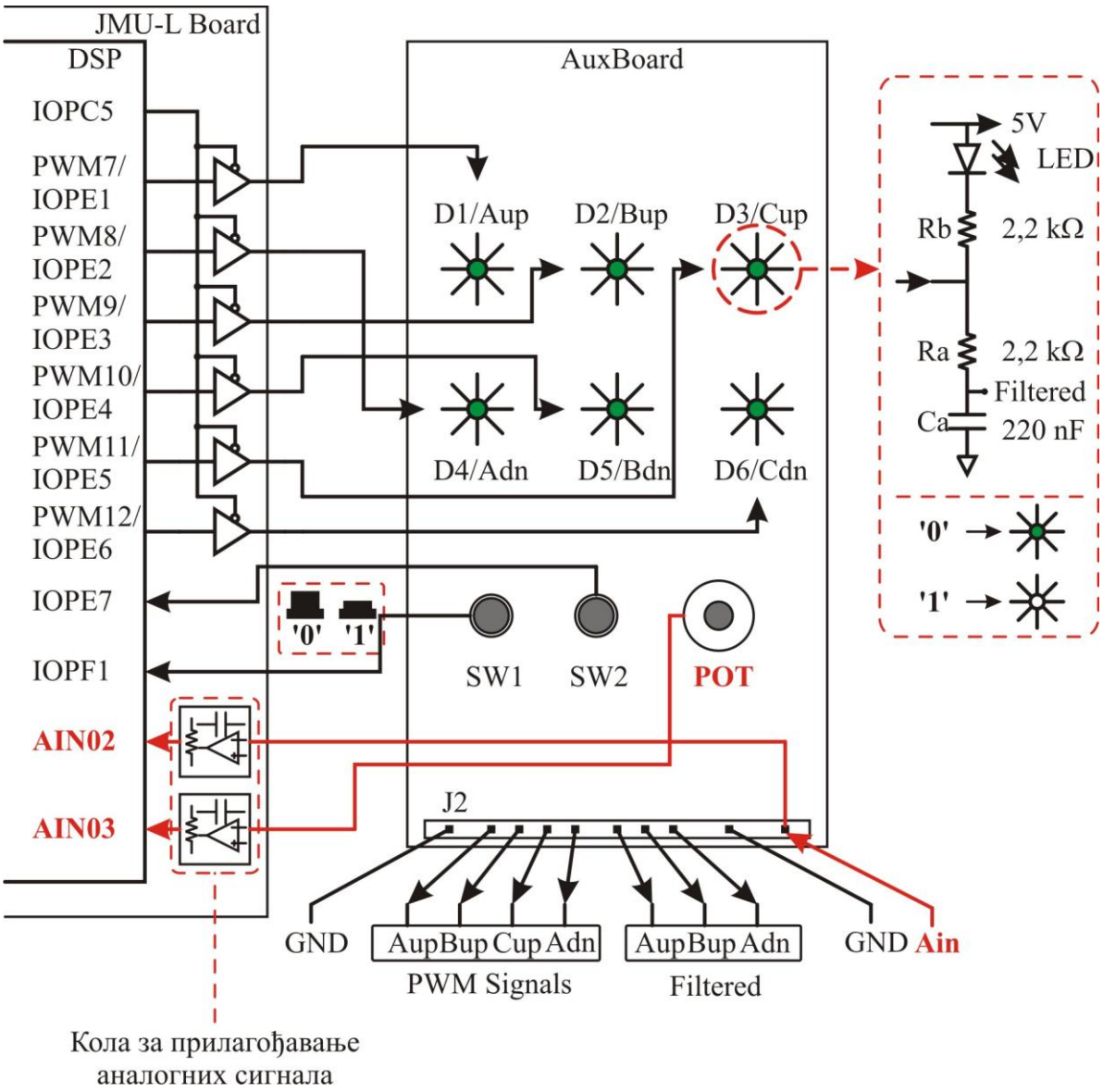
Hardverske veze i tok signala u ovoj vežbi prikazani su na Slici 2.9.

Pre nego što se analogni signal dovede na pin DSP-a koji predstavlja ulaz u A/D konvertor potrebno je taj signal obraditi odnosno prilagoditi za potrebe konverzije. To prilagođenje signala vrši posebni elektronski sklop (sastavljen od operacionih pojačavača, kondenzatora i otpornika) koji se zove kolo za prilagođavanje analognih signala, a njegova pozicija prikazana je šematski na Slici 2.9. Prilagođenje se sastoji u podešavanju naponskog nivoa signala na onaj koji odgovara ulazu DSP-a (0-3,3V), kao i njegove filtracije niskopropusnim filtrom. Nešto više reči o kolu za prilagođenje biće u poglavlju koje se bavi merenjem struje u pogonu.

Ovako prilagođen signal dovodi se na ulaz 10-bitnog A/D konvertora odnosno strukture koja je prikazana na Slici 2.11. Prilagođeni signal sa analognog ulaza Ain dovodi se na ulaz ADCIN02, a prilagođeni signal sa potenciometra na ulaz ADCIN03.U narednoj vežbi u kojoj će biti demonstriran rad A/D konvertora biće korišćen samo signal sa potenciometra.

Promenom otpornosti potenciometra menjaće se napon na ulazu ADCIN03 A/D konvertora i to u opsegu od -3,3 V do 3,3 V. Prolaskom kroz kolo za prilagođenje koje eliminiše šumove i skalira ovu vrednost napona, napon na ulazu ADCIN03 A/D konvertora variraće u opsegu od 0 do 3,3 V.

Nakon završene konverzije u registru se nalazi broj koji se nalazi u opsegu od 0 do 1023 i koji predstavlja rezultat konverzije odnosno digitalni ekvivalent analognog napona sa potenciometra. Sadržaj ovog registra će se kopirati u određenu promenljivu POT čija će vrednost zatim biti praćena na USB osciloskopu.

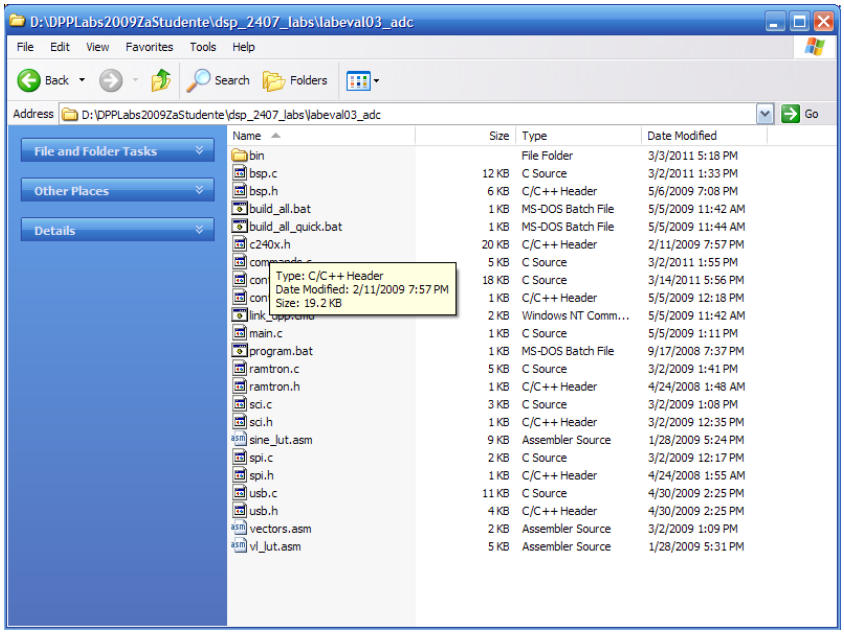


Slika 2.9 – Šematski prikaz veza za ADC vežbu

Tok vežbe

Radi uspešne izrade ove vežbe potrebno proći kroz sledeće korake:

1. Odabrati odgovarajući setap za izradu vežbe (ASIM1, ASIM2 i PMSM).
2. Uključiti PC računar i prekidač *main*.
3. Otvoriti projektni folder ***labeval03_adc*** koji se nalazi na putanji: *D:\DPPLabs2009ZaStudente\dsp_2407_labs\labeval03_adc*. Na Slici 2.12 prikazan je sadržaj projektnog foldera ***labeval03_adc***.



Slika 2.10 - Sadržaj projektnog foldera labeval03_adc

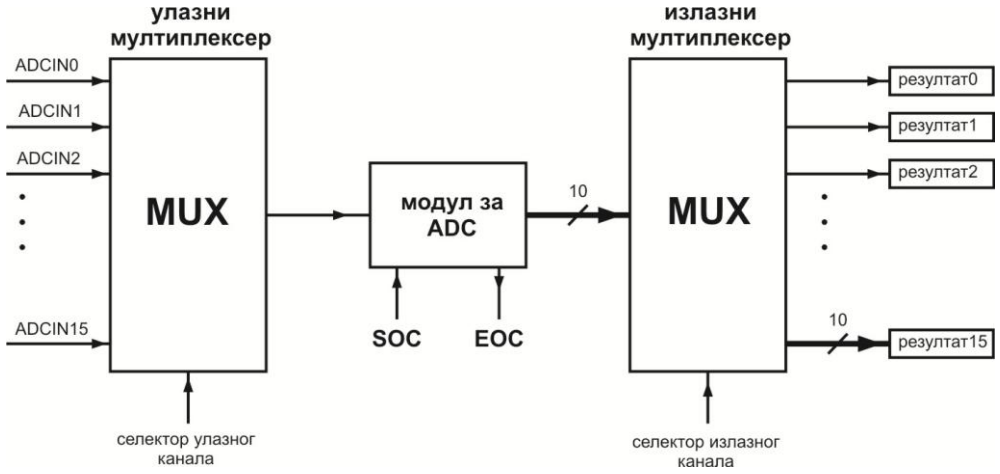
Fajl ***control.c*** sadrži glavne naredbe kojima se konfiguriše proces A/D konverzije.

4. Ponoviti proceduru kompajliranja programa i njegovog upisivanja u DSP koja je opisana u koraku 3 poglavlja **Demonstracija U/f upravljanja** u Demo vežbi.
5. Po završenom spuštanju programa u DSP, pritisnuti dugme *reset* na JMU-L ploči.
6. Sada otvoriti program *dspgui.exe* koji se nalazi na sledećoj putanji *D:\DPPLabs2009ZaStudente\dsp_2407_labs\dspgui*.
7. U otvorenom prozoru programa pritisnuti dugme *Connect* pa zatim *StartDAQ*.
8. U polje *Mnemonic* uneti ime promenljive POT i zatim kliknuti na dugme *Read.*
9. Ispod imena promenljive POT pojavljuje se vrednost rezultata AD konverzije.
10. Okretati potenciometar i pri svakom okretanju kliknuti na dugme *Read* i pratiti promenu rezultata AD konverzije.
11. Kliknuti na dugme *Disconnect* pa zatim zatvoriti prozor USB osciloskopa i isključiti prekidač *main.*
12. U okviru poglavlja Teorijske osnove dat je kraći opis koda i softverske realizacije ADC. Studenti koji nisu zainteresovani mogu preskočiti ovaj deo.

Teorijske osnove

- Procesor TMS320LF2407A je opremljen sa desetobitnim analogno/digitalnim konvertorom kome je pridruženo 16 ulaznih pinova ADCIN00-ADCIN15, kao i pinovi za dovođenje referentnog napona napajanja – VREFHI i masu VREFLO.
- Ovih 16 ulaza se vode na ulazni multiplekser koji je dalje priključen na modul za AD konverziju DSP-a (Slika 2.11). Posebnim programskim naredbama bira se kanal na multiplekseru koji vodi signal u modul za A/D konverziju (ADC) tj. time se bira signal sa odgovarajućeg ulaznog pina koji treba konvertovati. Upisivanjem odgovarajućeg broja u registar MAXCONV bira se broj signala koji se žele konvertovati (npr. ako se želi konverzija sa svih 16 kanala u navedeni registar treba upisati broj 15).
- AD konvertor se može konfigurisati tako da se nakon jednog starta konverzije obavi automatsko odabiranje svih 16 kanala i to proizvoljnim redosledom. Redosled se bira podešavanjem bitova SEQ registara (za detaljnije objašnjenje pogledati *TMS320LF/LC240xA DSP Controllers Reference Guide*, Texas Instruments). Potrebno je takođe podesiti da konverzija ne kreće automatski već kada se pojavi signal SOC (Start of Conversion – videti Sliku 2.11).

- Anlogno-digitalna konverzija se može obaviti na dva načina: kontinualno i pojedinačno. Kontinualna konverzija podrazumeva da se ulazni analogni signal kontinualno pretvara u digitalni ekvivalent bez zastoja, dok pojedinačna konvezija podrazumeva da se nakon jedne obavljene konverzije ne prelazi na sledeću dok se ne da signal za odobravanje. Regulisanje pojedinačne konverzije se najčešće vrši pomoću tajmera (brojača).



Slika 2.11 – Šematski prikaz A/D konvertora DSP-a

Start konverzije (engl. **S**tart-**o**f-**C**onversion - SOC) mogu zadati (u žargonu trigerovati, engl. trigger - okidač) menadžeri dagađaja, može se zadati softverski ili promenom naponskog nivoa na eksternom pinu, ADCSOC.

Rezultat A/D konverzije je 10-bitni broj koji se smešta u naročiti 16-bitni registar procesora levo poravnat kao što je prikazano na Slici 2.12. Pošto je opseg napona signala koji se dovodi na A/D konvertor 0-3,3V, rezolucija A/D konvertora je $3{,}3/1023 \approx 3{,}2mV$. Minimalno vreme trajanja konverzije jednog odbirka na TMS320LF2407A iznosi 500 ns.

ADC Result																
MSB									LSB							
b15	B14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	B3	b2	b1	b0	

Slika 2.12 – Izgled registra u koji se smešta rezultat A/D konverzije

Nakon konverzije bira se registar za smeštanje rezultata pomoću izlaznog multipleksera kao što je prikazano na Slici 2.12. To se takođe definiše određenim programskim naredbama. Po završetku konverzije može se generisati i prekid (signal EOC - End-of-Conversion) na osnovu koga korisnik može znati da je konverzija uspešno okončana.

Softver-kratak osvrt na kod

Na Slici 2.13 navedena je deklaracija promenljivih koje se koriste za AD konverziju.

```
// Promenljive vezane za A/D konverziju
volatile int adc_add00, adc_add01, adc_add02, adc_add03;
volatile int adc_read00, adc_read01, adc_read02,
adc_read03;
volatile int potenciometar;
```

Slika 2.13 - Definisanje promenljivih za potrebe AD konverzije

Za potrebe očitavanja vrednosti konverzije iz odgovarajućih registara i njihove obrade formirane su funkcije ***ADC_AddFirst*** i ***ADC_ReadFirst*** koje se nalaze u okviru glavnog programa ***control.c***. AD konverzija se vrši u interapt rutini procesora. Na svakih 25μs AD konvertor uzima po jedan odbirak analognog signala sa četiri analogna ulaza i smešta ih u odgovarajuće registare. Funkcijom ***ADC_AddFirst*** uzimaju se odbirci sa sva četiri kanala(koji se nalaze u odgovarajućim registrima) i nadodaju na prethodne odbirake,a rezultat se zatim smešta u odgovarajuće promenljive ***adc_add0x*** (x=0,1,2,3). Svakih 100 μs

izvršava se kontrolni algoritam i u okviru njega poziva se funkcija ***ADC_ReadFirst*** koja čita vrednost promenljive u kojoj se nalazi srednja vrednost prethodna četiri odbirka i ta vrednost se smešta u odgovarajuće promenljive ***adc_read0x*** (x=0,1,2,3). Promenljivoj ***potenciometar*** se zatim dodeljuje vrednost promenljive ***adc_read03*** koja predstavlja konvertovani signal sa potencionmetra. Na kraju se promenljiva ***potenciometar*** prikazuje na ekranu USB osciloskopa pod imenom POT. Iz ovoga se može zaključiti da se odbirci uzimaju mnogo češće nego što je to potrebno (četiri puta u 100 μs). Ovaj postupak se naziva *oversempling* i koristi se kao metod za filtraciju struje motora koji se napaja iz pretvarača (invertora). Taj proces u ovom primeru nije neophodan ali je implementiran da bi se student na vreme upoznao sa ovim pojmom. *Oversempling* će detaljnije biti objašnjen u poglavlju koje se bavi merenjem struje. Na Slici 2.14 prikazan je kod funkcija ***ADC_AddFirst*** i ***ADC_ReadFirst.*** U ovom trenutku se neće ulaziti detaljnije u način implementacije samih funkcija već je dovoljno znati čemu one služe.

```
//*****************************************************************************
//Ova funkcija se koristi prilikom ocitavanja vrednosti sa potenciometra
// na GUI-u (Data)
//*****************************************************************************

void ADC_AddFirst( void )
{
        adc_add00 += RESULT0 >> 6; adc_add01 += RESULT1 >> 6;
        adc_add02 += RESULT2 >> 6; adc_add03 += RESULT3 >> 6;

        ADCTRL2 |= ( SET6 | SET14 );                // Reset Sequencer
        ADCTRL2 |= SET13;                           // Start Conversion
}

//*****************************************************************************
//Ova funkcija se koristi prilikom ocitavanja vrednosti sa potenciometra
// na GUI-u (Data)
//*****************************************************************************

void ADC_ReadFirst( void )
{
        adc_read00 = adc_add00; adc_read01 = adc_add01;
        adc_read02 = adc_add02;         adc_read03 = adc_add03;

        adc_add00 = adc_add01 = adc_add02 = adc_add03 = 0;
}
```

Slika 2.14 - Programski kod funkcija **ADC_AddFirst** i **ADC_ReadFirst**

Zadaci i pitanja

1. Koliko traje konverzija jednog odbirka na MOT09 platformi sa 2407?
2. Da li je moguće ili nije moguće organizovati (konfigurisati) AD registre tako da se nakon jednog starta konverzije dogodi automatsko odabiranje svih 16 kanala, i to sa rasporedom, prvo svi parni, a posle svi neparni ulazi? A u standardnom redosledu od prvog pa redom do poslednjeg?
3. U vezi prethodnog pitanja, koliko bi trajala automatska konverzija svih 16 izlaza ako se konvertuju redom, od prvog do poslednjeg?
4. Ako je ulazni napon 1 volt, koji broj će se naći u ADC Result?
5. Nabrojati tri načina na koji se može započeti (inicirati) konverzija.
6. Na koji način korisnik može biti siguran da je komandovana konverzija okončana?
7. Koliko vremena pre početka konverzije se mora zatvoriti prekidač S/H kola i na koji način se ostvaruje zatvaranje i otvaranje ovog prekidača?

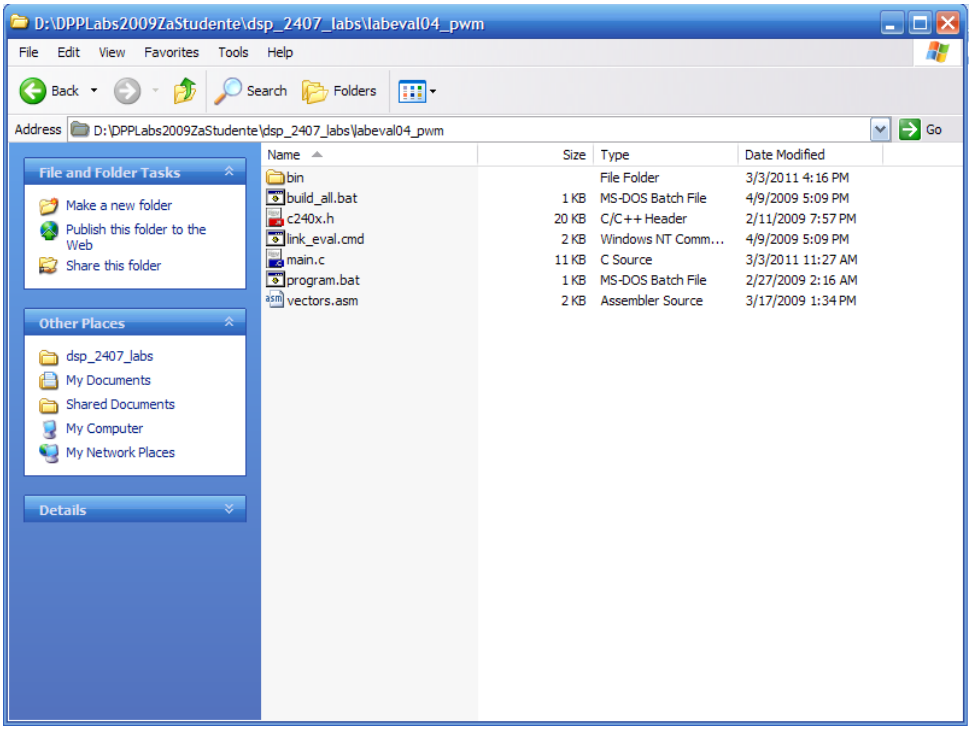
2.6 VEŽBA 4 - Modul za generisanje PWM impulsa

Cilj vežbe

- Student će se u ovoj vežbi upoznati sa pojmom impulsno-širinske modulacije i radom PWM modula DSP-a.
- U vežbi će kao modulišući signal biti korišćen jednosmerni napon sa potenciometra.
- Taj jednosmerni napon će se konvertovati u digitalni ekvivalent koji će biti smešten u CMPR4 compare registar procesora. Vrednost tog napona menja će se okretanjem potenciometra čime će se menjati vrednost u CMPR4 registru.
- Promenom vrednosti u CMPR4 registru procesora menjaće se i širina impulsa koji nastaju kao rezultat poređenja njegovog sadržaja i nosioca što je u ovom slučaju vrednost brojača-tajmera. Na ovaj način se ustvari menja indeks modulacije.
- Ovi impulsi se dalje vode na dve diode AuxBoard pločice (koje predstavljaju jednu granu invertora) i na letvu sa mernim mestima. Na letvi sa mernim mestima biće prikačen osciloskop na kome će student moći da prati promenu širine impulsa okretanjem potenciometra. Takođe promena širine impulsa menja i jačinu svetlosti koju emituju diode.
- Vežba počinje sa poglavljem **Tok vežbe**, dok je teorijsko objašnjenje impulsno-širinske modulacije dato u poglavlju **Teorijske osnove** koje se preporučuje zainteresovanim studentima radi produbljivanja znanja. U poglavlju **Softver-kratak osvrt na kod** dat je opis glavnih softverskih naredbi uz pomoć kojih se upravlja PWM modulom.

Tok vežbe

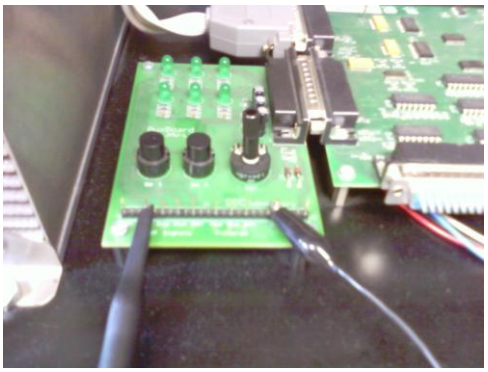
1. Odabrati odgovarajuću postavku za izradu vežbe (ASIM1, ASIM2 i PMSM).
2. Uključiti PC računar i prekidač *main*.
3. Otvoriti projektni folder ***labeval04_pwm*** koji se nalazi na sledećoj putanji *D:\DPPLabs2009ZaStudente\dsp_2407_labs\labeval04_pwm*, a čiji je sadržaj prikazan na Slici 2.15.



Slika 2.15 - Sadržaj projektnog foldera labeval04_pwm

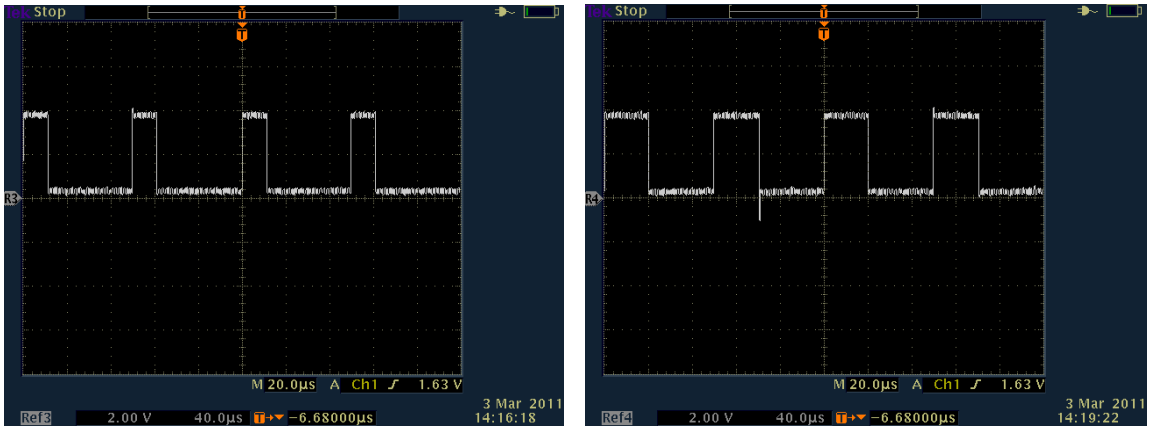
Fajl ***main.c*** sadrži glavne naredbe kojima se konfiguriše PWM modul DSP-a. Najvažniji deo ovog programa analiziran je poglavlju **Softver-kratak osvrt na kod**.

4. Ponoviti proceduru kompajliranja programa i njegovog upisivanja u DSP koja je opisana u koraku 3 poglavlja **Demonstracija U/f upravljanja** u Demo vežbi.
5. Po završenom spuštanju programa u DSP, pritisnuti dugme *reset* na JMU-L ploči.
6. Sada je potrbno prikačiti sondu osciloskopa na letvu sa mernim mestima i to na pin Aup, a uzemljenje na GND prema slici 2.16.



Slika 2.16 - Prikačena sonda na letvu sa mernim mestima

7. Uključiti osciloskop. Za dalje podešavanje osciloskopa obratiti se dežurnom asisentu.
8. Okretati potenciometar i posmatrati promenu širine impulsa na osciloskopu. Na Slici 2.17 prikazan je izgled povorke PWM impulsa na ekranu osciloskopa za dva različita položaja potenciometra.



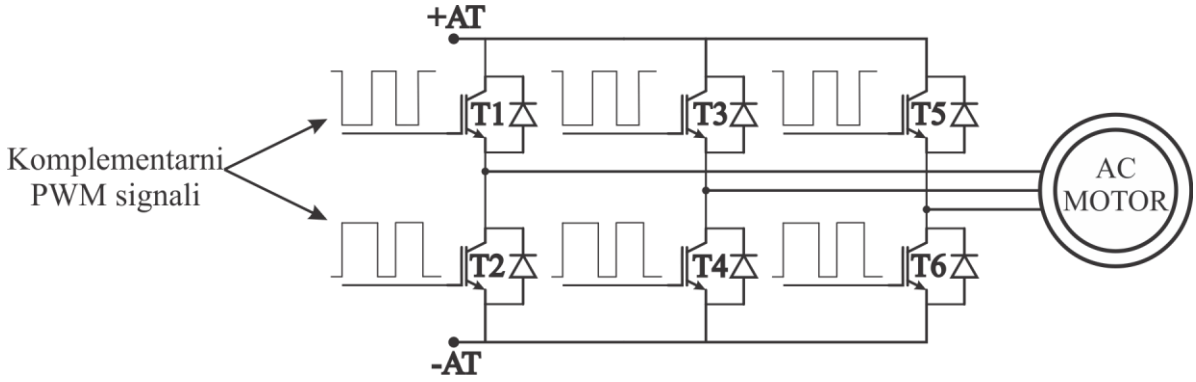
Slika 2.17 - Povorka PWM za dva različita indeksa modulacije

9. Obratiti pažnju i na promenu jačine svetlosti koju emituju diode na AuxBoard pločici.
10. Isključiti osciloskop, a zatim skinuti sondu.
11. Isključiti prekidač main.
12. Setap je spreman za narednu vežbu.
13. Slede poglavlja **Teorijske osnove** i **Softver-kratak osvrt na kod** u kojima je ukratko opisan princip impulsno-širinske modulacije i PWM modul DSP-a kao i osnovne naredbe kojima se ovaj modul podešava. Ova poglavlja nisu neophodna za dalju izradu vežbe, a preporučuje se zainteresovanim studentima radi produbljivanja znanja.

Teorijske osnove

Optimalno upravljanje trofaznim naizmeničnim motorom je moguće samo ako postoji mogućnost kontinualne promene amplitude i frekvencije statorskog napona. To se ostvaruje primenom digitalno upravljanog trofaznog tranzistorskog invertora (u daljem tekstu samo invertor). Svakom od tranzistora u

invertoru šalje se povorka impulsa koji predstavljaju signale za njegovo paljenje odnosno gašenje. Topologija motora koji se napaja invertorom data je na Slici 2.18.



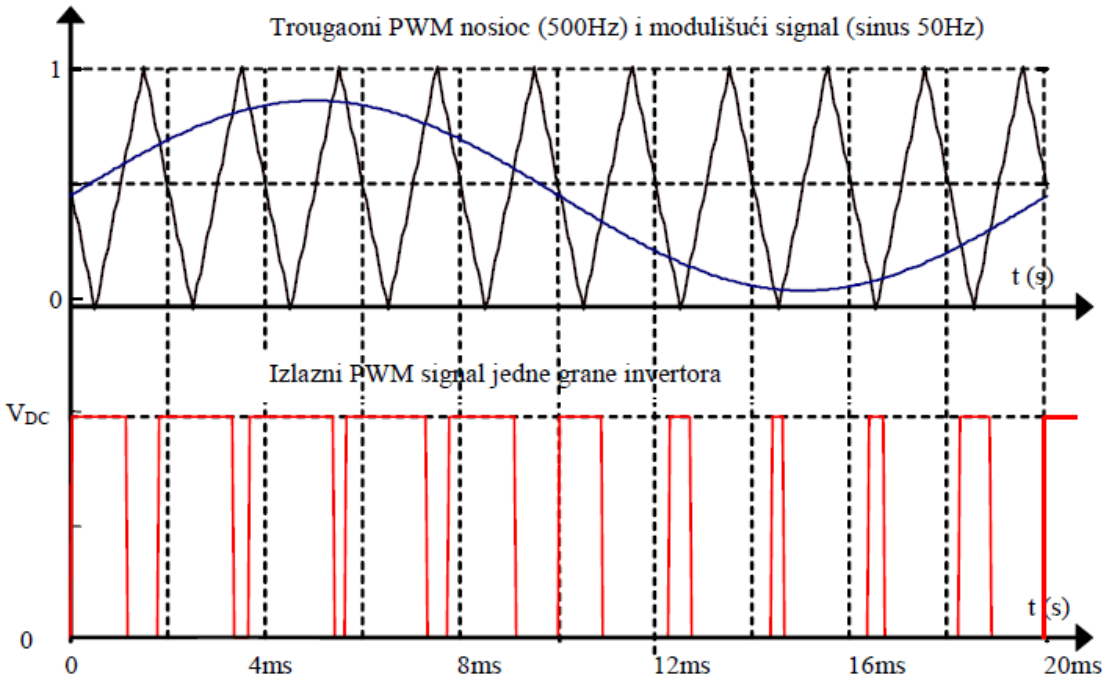
Slika 2.18 - Topologija invertorski napajanog trofaznog motora

Kontrolisanjem signala za paljenje tranzistora upravlja se njihovim vremenom vođenja odnosno samim tim širinom pravougaonih naponskih impulsa koji se dovode na motor. Menjanjem širine naponskih imulsa menja se srednja vrednost dovedenog napona na motor. Ova promena širine naponskih impulsa koji se vode na motor naziva se impulsno širinska modulacija (IŠM, engl. **P**ulse **W**idth **M**odulation - **PWM**). PWM u stvari pretstavlja reprezentaciju analognih signala digitalnom aproksimacijom i sastoji iz povorke impulsa promenljive širine koja je proporcionalna trenutnoj vrednosti analognog signala koji treba da bude predstavljen. Ova povorka impulsa se dobija na sledeći način. Kao prvo potrebno je imati dva signala:

Modulišući signal m(t), predstavlja analogni signal koji treba da bude dočaran uz pomoć IŠM.

Nosilac n(t), predstavlja signal visoke učestanosti koji će biti modulisan. On najčešće ima oblik testere, a njegova učestanost je reda 10kHz.

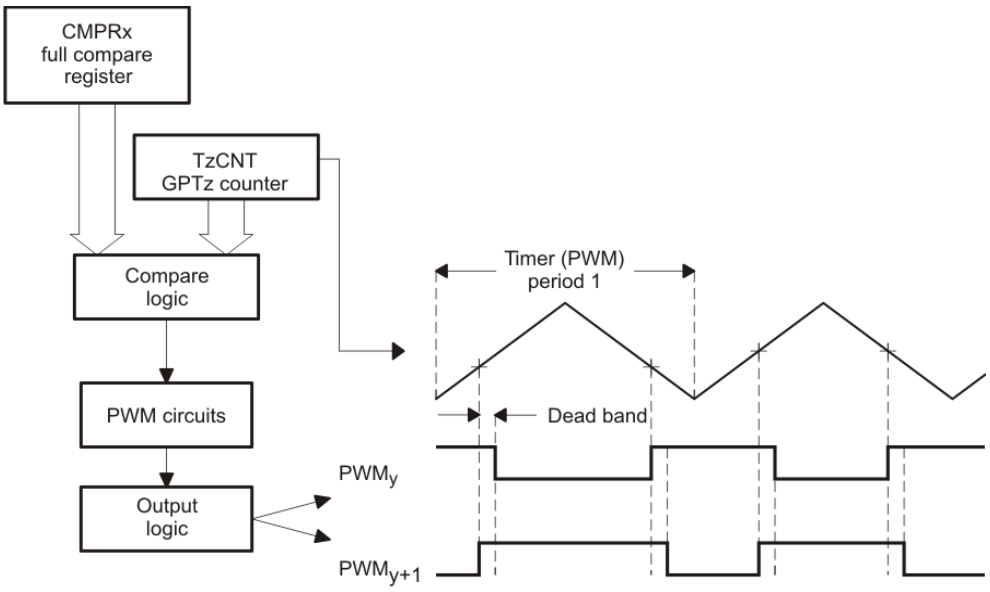
Korišćenjem compare registara DSP-a vrši se stalno poređenje ova dva signala. U trenutku kada modulišući signal postane veći od nosioca generiše se impuls za paljenje tranzistora, a kada postane manji generiše se impuls za njegovo gašenje. Vremenski razmak između ova dva trenutka predstavlja vreme vođenja tranzistora i označava se sa t_{on}. Ukupno vreme koje protekne između dva paljenja tranzistora označava se sa T. Odnos ova dva vremena predstavlja indeks modulacije u oznaci $m = t_{on}/T$. U zavisnosti od izgleda modulišućeg signala razlikujemo više vrsta PWM modulacija. Najčešće korišćena sinusna modulacija prikazana je na Slici 2.19 na kojoj se mogu uočiti modulišući sinusni signal, testerasti nosilac kao i povorka PWM impulsa. Sa ove slike se jasno vide trenutci paljenja i gašenja tranzistora u jednoj grani invertora. Što je učestanost nosioca veća vreme T je kraće pa se srednja vrednost dobijenog PWM signala potpuno približava trenutnoj vrednosti modulišućeg signala.



Slika 2.19 - Prikaz sinusne modulacije

Ispravno je na kraju postaviti pitanje: Da li će ovakav napon na motoru proizvesti isto ponašanje (kretanje) motora kao u slučaju sinusoidalnog napajanja? Impulsi koje dovodimo na motor su konstantne amplitude i sadrže (približno) istu količinu energije kao i originalni analogni signal. Ova osobina je veoma važna u digitalnoj kontroli motora i upravljanju kretanjem jer sinusoidalna struja (energija) može biti dostavljena motoru preko invertora pomoću opisanih PWM signala. Iako se energija motoru dostavlja u diskretnim paketima, mehanička inercija rotora motora se ponaša kao niskoprousni filtar (engl. smoothing filter) pa je dinamika kretanja motora iz tog razloga slična sa dinamikom koju bi imali u slučaju primene sinusoidalnih struja na motor. Ako bi povorku PWM signala predstavili preko spektra, visokofrekventne komponente spektra ne bi imale nikakav uticaj na motor već bi njegovo kretanje prouzrokovala samo niskofrekventna komponete učestanosti modulišućeg signala što je ono što je i potrebno. Za više detalja o PWM modulaciji pogledati predavanja prof Slobodana Vukosavića iz predmeta DPP1. Sada će ukratko biti izložen način realizacije PWM pomoću DSP-a.

DSP poseduje zasebne module za generisanje PWM signala, a to su ustvari ranije pomenuti menadžeri događaja. Svaki menadžer događaja u procesoru TMS320LF2407A poseduje po tri jedinice za poređenje – compare units (grupisane u full compare units), a svaka od ovih jedinica generiše po dva komplementarna PWM izlaza. DSP je namenjen za kontrolu motora naizmenične struje – brushless motora. Svaka grana invertora sadrži po dva energetska tranzistora (Slika 2.18) koja se kontrolišu sa dva komplementarna PWM signala. Svaki compare unit daje dva komplentarna PWM signala koja se vode na po jednu granu trofaznog tranzistorskog invertora.



EV	Time base (Timer *z*)	Compare reg. *x*	Outputs ($PWM_{y,y+1}$)
EVA	Timer 1	CMPR1	1, 2
		CMPR2	3, 4
		CMPR3	5, 6
EVB	Timer 3	CMPR4	7, 8
		CMPR5	9, 10
		CMPR6	11, 12

Slika 2.19 - Principijelna blok šema logike za generisanje PWM signala sa simetričnim nosiocem

Principijelna blok šema modula za generisanje PWM signala procesora je prikazana na Slici 2.19. Da bi se generisao PWM signal, potreban je signal nosilac (vremenska baza, engl. time-base signal), odnosno tajmer opšte namene čiji je period jednak periodu PWM signala. U slučaju menadžera događaja A, za generisanje signala nosioca koristi se tajmer 1, a u slučaju menadžera događaja B tajmer 3. Perioda PWM signala jednaka periodu tajmera, 2*TzPER, z = 1, 3. Vrednost tajmera (u registru TzCNT, z = 1 - EVA, z = 3 - EVB) se konstantno poredi sa vrednostima u registrima za poređenje – compare registers. U slučaju menadžera događaja A to su registri CMPR1 – CMPR3, a u slučaju menadžera događaja B su u pitanju registri CMPR4 – CMPR6. U ovaj vežbi se koristi menadžera događaja B, a nosilac je simetričan (trougaoni). Događaj koji se dešava kada se neki od registara za poređenje izjednači sa trenutnom vrednoćšu tajmera naziva se compare (compare match) i tada se dešava tranzicija odgovarajućih PWM izlaza (0→1 ili 1→0) u zavisnosti od trenutnog stanja tajmera. Dužina trajanja impulsa je proporcionalna vrednosti koja se nalazi u odgovarajućem compare registru. Dakle, kada se desi compare događaj, logika za poređenje (compare logic) daje aktivacioni signal kolima za generisanje PWM signala (PWM circuits) koja preko izlazne logike (output logic) formira PWM signale.

U situaciji kada kada su oba gejta tranzistora u istoj grani invertora istovremeno otvorena dovodi napajanje u kratak spoj i rezultuje pojavom velike struje kroz tranzistore. Problem se javlja jer se tranzistori tipično brže pale nego što se gase, a sa druge strane jer se gornji i donji tranzistor u grani naizmenično pale i gase (komplementarni su). Mrtvo vreme obezbeđuje potpuno gašenje jednog tranzistora pre nego što se upali drugi tranzistor i određeno je odgovarajućim karakteristikama upotrebljenih energetskih tranzistora u konkretnoj aplikaciji. Iako je trajanje ove struje kratkog spoja konačno u toku jednog PWM perioda, čak i veoma kratki periodi izlaganja tranzistora ovoj struji mogu izazvati veoma veliko zagrevanje i preopterećenje poluprovodničkih komponenata. Svaka od jedinica za poređenje DSP-a sadrži i logiku za generisanje mrtvog vremena (engl. *dead-band*) što rešava prethodno opisani problem. Ovo logičko kolo se konfiguriše pomoću ACTR registra. *Dead-band* se može nezavisno podešavati za svaku od jedinica za poređenje i to u opsegu od 0-12 μs.

Softver-kratak osvrt na kod

Na Slici 2.20 prikazan je deo koda glavne funkcije ***main.c*** u kome se nalazi interapt rutina DSP-a (linije koda od 302 do 319). U interapt rutini DSP izvršava redom sledeće instrukcije:

(1) Poziva funkcije za AD konverziju kao u vežbi za AD konverziju (poglavlje 2.5). Analogni napon sa potenciometra se na ovaj način pretvara u broj, a rezultat se smešta u promenljivu ***adc_read03***.

(2) ***if*** naredba u kojoj se proverava da li je rezultat konverzije tj. napon sa potenciometra veći od neke definisane granice (u ovom slučaju 2000, što odgovara naponu od oko 3V). Ova granica je određena najvećom vrednošću nosioca tj. tajmera sa kojom se vrši poređenje.

(3) U koliko je vrednost konvertovanog napona odnosno promenljive ***adc_read03*** veći od 2000 u registar za poređenje ***CMPR4*** upisuje se vrednost 2000

(4) U koliko je manja u registar ***CMPR4*** upisuje se vrednost promenljive ***adc_read03***

Iz ovoga se vidi da je u navedenoj ***if*** naredbi moguće definisati granicu po želji i na taj način ograničiti maksimalnu širinu impulsa tj. indeks modulacije.

```
      if ( int2_source == INT2_T1PINT )
              {
                      // First interrupt: period expired
(1)                   ADC_AddFirst();
                      ADC_ReadFirst();

(2)                   if ((adc_read03>>1)>2000)
                              {
(3)                                   CMPR4 = 2000;
                              }
                      else
                              {
(4)                                   CMPR4 = adc_read03>>1;
                              }

                      int2_which_int = 0;
              }
```

Slika 2.20 - Deo koda ukome se definiše vrednost registra za poređenje odnosno indeksa modulacije

Zadaci i pitanja

1. U delu koda funkcije ***main.c*** koji je prikazan na Slici 2.18 napraviti potrebne izmene tako da indeks modulacije bude ograničen na 0,5.
2. Objasniti zbog čega pri okretanju potenciometra dolazi do promene jačine svetlosti koju emituju diode.

Napomena: *nakon načinjenih izmena, sačuvati fajl* ***main.c*** *pa zatim ponoviti korake 1-7 iz poglavlja* ***Tok vežbe*** *vežbe 4.*

3. Kako će se promeniti srednja vrednost napona ako se brojanje brojača promeni sa Up-Down na samo Up?
4. Kako će izgledati naponski impulsi ako je u CMPR registar upisan broj veći od broja u periodnom registru?
5. Kako će izgledati naponski impulsi ako je u CMPR registar upisan broj manji od nule, to jest broj koji počinje jedinicom kao MSB?
6. Šta je dead band na slici 2.19, čemu služi i kako se podešava?
7. Koja je rezolucija u zadavanju napona kod trofaznog invertora, izražena u voltima za konkretnu postavku?

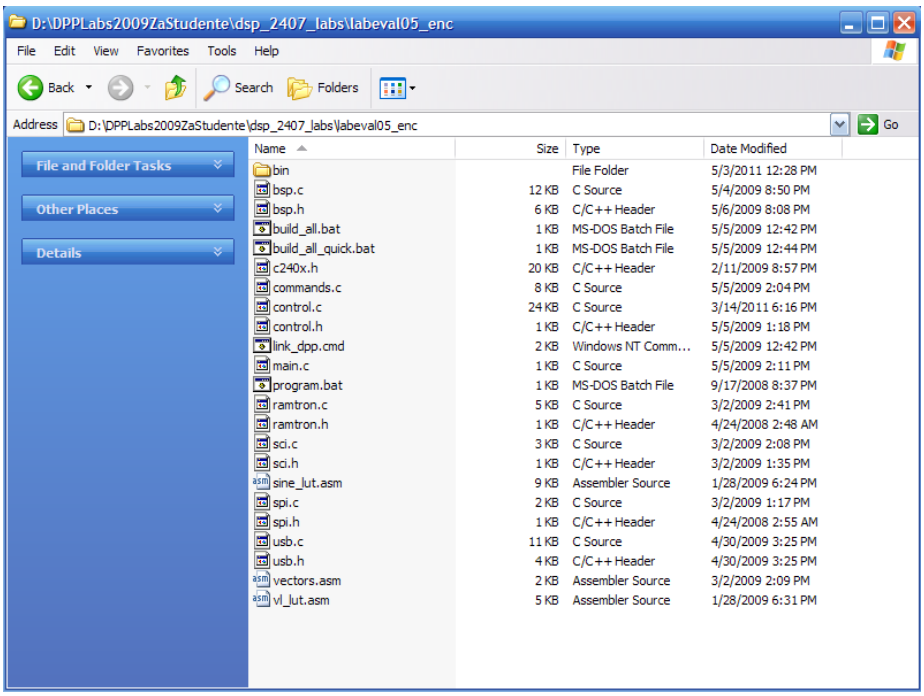
2.7 VEŽBA 5 - Enkoderski interfejs

Cilj vežbe

- Upoznavanje studenta sa načinom merenja pozicije odnosno brzine motora pomoću enkodera, kao i sa modulom DSP-a za obradu enkoderskih impulsa.
- Prikazivanje sadržaja brojača DSP-a u koji se smešta rezultat obrade kvadraturnih enkoderskih impulsa odnosno apsolutna pozicija rotora motora.
- Vežba je demonstracionog karaktera, a počinje sa poglavljem **Tok vežbe**. U poglavlju „6.10 Dodatak 3 – Merenje pozicije“ izložne su osnovne informacije o merenju pozicije i načinu pretvaranja enkoderskih impulsa u apslolutnu poziciju rotora motora.

Tok vežbe

1. Odabrati odgovarajući setap za izradu vežbe (ASIM1, ASIM2 i PMSM).
2. Uključiti PC računar i prekidač *main*.
3. Otvoriti projektni folder ***labeval05_enc*** koji se nalazi na putanji *D:\DPPLabs2009ZaStudente\dsp_2407_labs\labeval05_enc*. Na Slici 2.21 prikazan je sadržaj projektnog foldera ***labeval05_enc***.



Slika 2.21 - Sadržaj projektnog foldera labeval04_pwm

4. Ponoviti proceduru kompajliranja programa i njegovog upisivanja u DSP koja je opisana u koraku 3 poglavlja **Demonstracija U/f upravljanja** u Demo vežbi.
5. Po završenom spuštanju programa u DSP, pritisnuti dugme *reset* na JMU-L ploči.
6. Uključiti prekidač power
7. Nakon toga treba pokrenuti program *dspgui.exe* koji se nalazi u istoimenom folderu na putanji *D:\DPPLabs2009ZaStudente\dsp_2407_labs\dspgui.*
8. Čekirati kanal 0 na gornjem ekranu USB osciloskopa.
9. Kliknuti na dugme *Connect* pa zatim na dugme *TurnOn.*
10. U polje *mnemonic* uneti WR, a u polje *Data 100* pa zatim kliknuti na dugme *Write.*
11. Nakon ovoga na ekranu se pojavljuje testerasta funkcija koja predstavlja promenu pozicije rotora (Slika 2.22). Broj 8000 odgovara uglu od 2π radijana.

Slika 2.22 - Pozicija rotora na ekranu USB osciloskopa

12. Sada klikuti na dugme *TurnOff* pa zatim na *Disconnect* pa isključiti prekidače *main* i *power*.

13. Ovim je završena ova demonsraciona vežba.

Više o načinu merenja pozicije na laboratorijskim postavkama može se naći u ovom uputstvu i to u poglavlju 6.10 Dodatak 3 – Merenje pozicije.

Zadaci i pitanja

1. Koje registre treba konfigurisati da bi se mogla čitati pozicija enkodera čiji su signali dovedeni na QEP1-2?
2. U kom registru će se akumulirati inkrementi pozicije rotora?
3. Koliko će se promeniti pomenuti registar ako se rotor pomeri za 1/4 obrtaja, a enkoder ima N impulsa po obrtaju?
4. Ako se rotor obrće brzinom 3000 o/min, koliki je inkrement pročitane pozicije rotora u 100 mikrosekundi?
5. Šta će se dogoditi ako se ukrste faze enkodera A i B?(Pogledati predavanja prof.dr Slobodana Vukosavića)
6. Dati formulu po kojoj se može izračunati brzine obrtanja motora u radijanima u sekundi.
7. Kako treba povezazi (tj. na koji pin) C-marker enkodera?

3. LABORATORIJSKA VEŽBA 3 - U/f upravljanje

Cilj vežbe

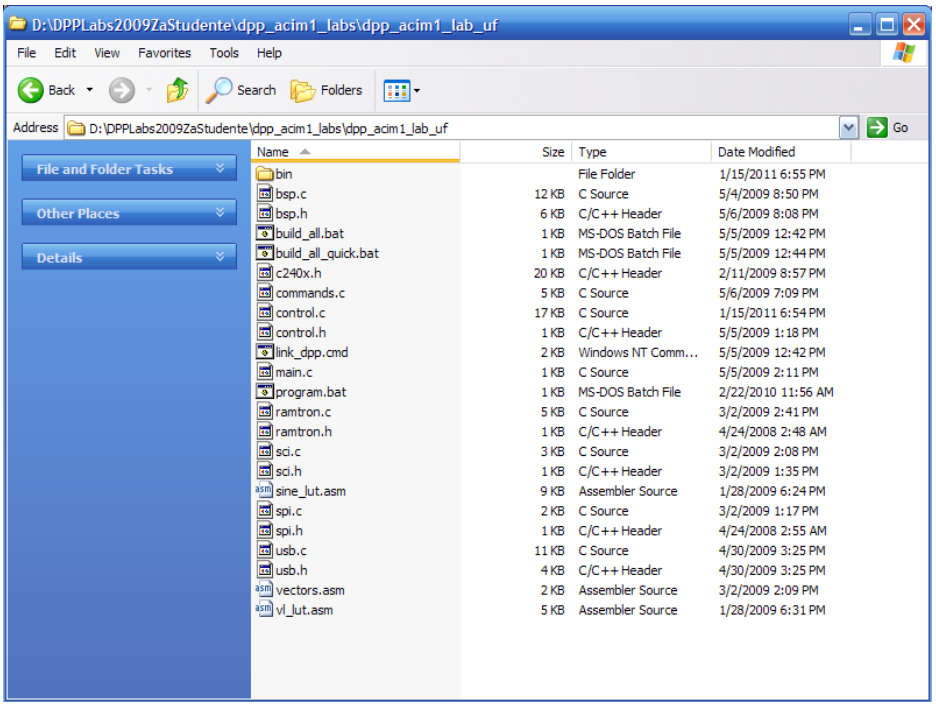
- Upoznavanje studenta sa osnovnim principima U/f upravljanja
- Upisivanje u DSP programa kojim se vrši U/f upravljanje
- Zadavanje referentne frekvencije preko USB osciloskopa na ekranu računara i prikaz rampe po kojoj se uspostavlja zadata referentna frekvencija.
- Upoznavanje studenta sa pojedinim programskim naredbama kojima se određuje napon koji se dovodi na faze motora, inkrement pozicije, kao i postupak transformacije napona u_{dq} u napone u_{abc}.
- Prikaz faznih struja motora u procesu zaletanja i u ustaljenom stanju.

Tok vežbe

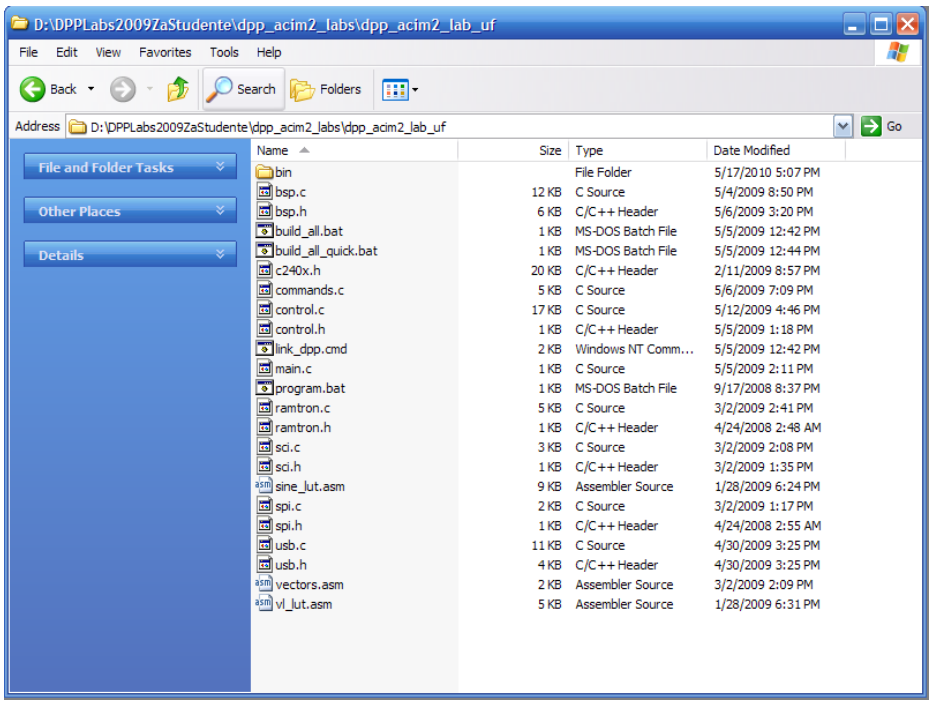
1. Odabrati odgovarajuću potavku za izradu vežbe (ASIM1, ASIM2).

2. Uključiti PC računar i prekidač *main*.

3. Otvoriti projektni folder *dpp_acim1_lab_uf* odnosno *dpp_acim2_lab_uf* koji se nalaze na sledećim putanjama *D:\DPPLabs2009ZaStudente\dpp_acim1_labs\ dpp_acim1_lab_uf* i *D:\DPPLabs2009ZaStudente\dpp_acim2_labs\dpp_acim2_lab_uf*, a čiji su sadržaji prikazani na Slici 3.1 i Slici 3.2.



Slika 3.1 - Sadržaj foldera dpp_acim1_lab_uf



Slika 3.2 - Sadržaj foldera dpp_acim1_lab_uf

- Fajl ***control.c*** sadrži naredbe koje omogućavaju implementaciju U/f upravljanja. Najvažniji deo ovog programa analiziran je poglavlju **Softver-kratak osvrt na kod**.

4. Ponoviti proceduru kompajliranja programa i njegovog upisivanja u DSP koja je opisana u koraku 3 poglavlja **Demonstracija U/f upravljanja** u Demo vežbi.

5. Po završenom spuštanju programa u DSP, pritisnuti dugme *reset* na JMU-L ploči.

6. Uključiti prekidač power

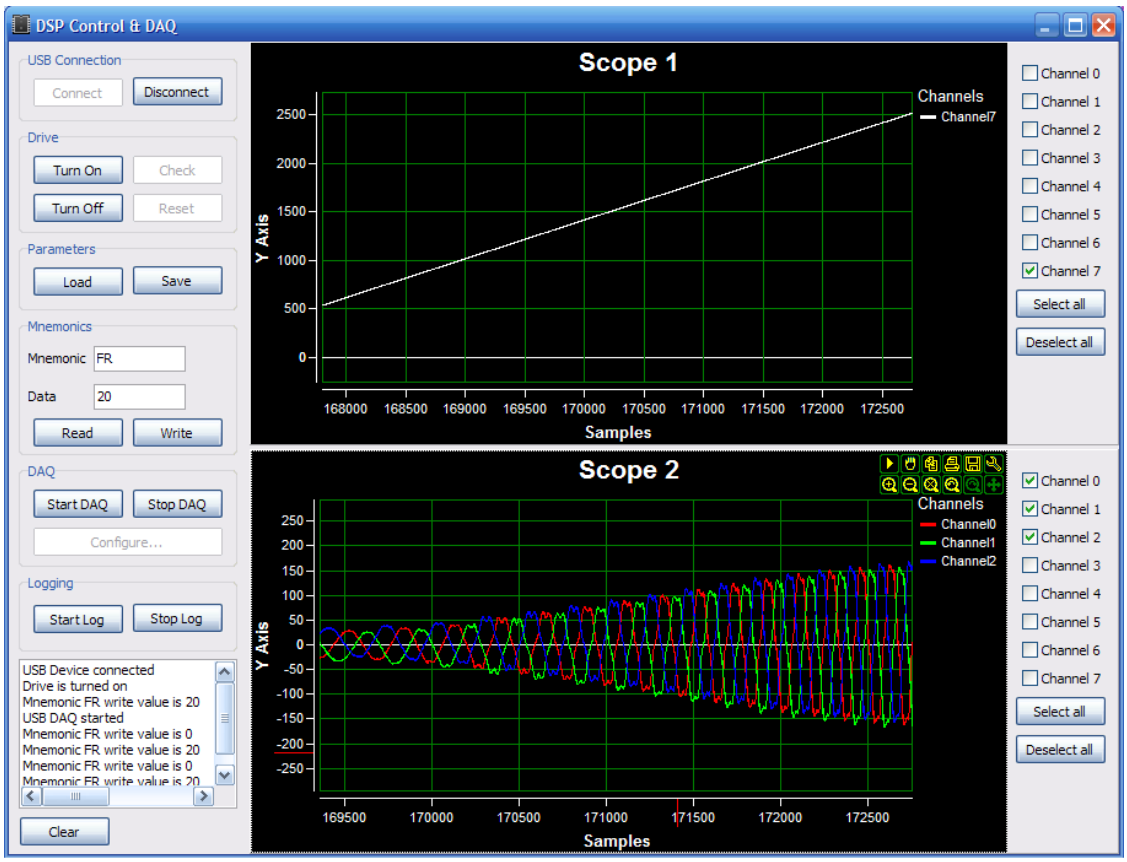
7. Nakon toga traba pokrenuti program *dspgui.exe* koji se nalazi u istoimenom folderu na putanji *D:\DPPLabs2009ZaStudente\dpp_acim1_labs\dspgui.*

8. Čekirati kanal 7 na gornjem ekranu i kanalale 0, 1 i 2 na donjem ekranu USB osciloskopa.

9. Kliknuti na dugme *Connect* pa zatim na dugme *TurnOn.*

10. U polje *mnemonic* uneti FR a u polje pored labele *Data* broj ***20***. Kliknuti na dugme *Write* čime je zadata referentna frekvencija faznog napona $f_r = 20[Hz]$.

11. Nakon ovoga motor bi trebao početi da se obrće. Brzina se ne menja naglo jer se referentna frekvencija menja po rampi do zadate vrednosti i taj prikaz se pojavljuje na gornjem ekranu USB osciloskopa. Na donjem ekranu se vide fazne struje motora koje su prikazane i na slici 3.3 zajedno sa uspostavljanjem brzine motora.



Slika 3.3 – Fazne struje motora pri njegovom zaletanju

Komentar: Sa Slike 3.3 se uočava da se pri zaletanju motora sa U/f kontrolom amplitude sve tri fazne struje menjaju po rampi do uspostavljanja zadate brzine za razliku od drugih metoda puštanja motora u rad gde se javlja problem velike polazne struje.

12. Sada kliknuti na dugme *TurnOff* pa zatim na *Disconnect* pa isključiti prekidače *main* i *power*.

13. Ovim je završena vežba U/f upravljanja. Sledi poglavlje **Teorijske osnove** u kojem je ukratko opisan princip U/f upravljanja.

Teorijske osnove

Kod napajanja iz trofaznog tranzistorskog invertora sa impulsno širinskom modulacijom, učestanost ω_S osnovne komponente napona (fundamentala) određuje sinhronu brzinu, dok količnik amplitude statorskog napona i učestanosti napajanja određuje fluks mašine. Promenom učestanosti napajanja asinhrone mašine menja se i sinhrona brzina Ωs.

U/f upravljanje asinhronim motorom se zasniva na kontroli amplitude i frekvencije napona na njegovim priključnim krajevima. Zadavanjem odgovarajuće statorske frekvencije računa se amplituda napona na osnovu zakona U/f=const.

Prostoperiodični signal se menja u vremenu. Da bi se izračunalo vreme vođenja tranzistora neophodno je da se zna vremenski trenutak u kome se generiše PWM impuls. Potrebno je znati novu vrednost ugla koja se kontroliše i čija brzina promene zavisi od zadate frekvencije. Svakom novom PWM periodu odgovara diskretni ugaoni položaj. Trenutni ugao u kT_{PWM} intervalu ima vrednost $\theta(kT_{PWM}) = \omega_s kT_{PWM} = 2\pi k \frac{f_s}{f_{PWM}}$.

Detaljanije objašnjenje U/f kontrole može se naći u predavanjima iz predmeta DPP1 prof. S.Vukosavića.

Softver-kratak osvrt na kod

Na Slici 3.4 prikazan je S kod funkcije kojom se implementira U/f regulacija. Ovaj kod je deo glavnog programa ***control.c.*** U nastavku će biti opisane glavne naredbe i delovi koda sa slike 3.4.

(1)-(4) Ovaj deo koda detaljno je opisan u poglavlju 1, potpoglavlje **Demonstracija U/f upravljanja.**

(5) Određivanje ugaonog položaja rotora i implementacija čitanja sinusne i kosinusne tabele u cilju određivanja sinusa i kosinusa ugaonog položaja

(6) U slučaju kada je zadata referentna frekvencija statorskog napona manja od nominalne frekvencije, statorski napon se određuje na osnovu zadate frekvencije i poznatog faktora skale u skladu sa zakonom upravljanja U/f=const. Ako je zadata frekvencija po apsolutnoj vrednosti veća od nominalne, tj. ako se od motora zahteva da se obrće brzinom većom od sinhrone, naponu napajanja se dodeljuje nominalna vrednost jer napon ne sme uzimati vrednosti veće od nominalne. Ako je zadata frekvencija veća od nominalne i manja od nule, tj. ako se zahteva obrtanje u suprotnom smeru od referentnog, naponu napajanja se dodeljuje negativna nominalna vrednost.

(7) Izračunavanje α i β komponenti statorskog napona.

Značenje pojedinih promenljivih:

1. ***drive1_f_ref*** – referentna vrednost frekvencije koja se zadaje preko GUI-a
2. ***drive1_f –*** frekvencija koja se uspostavlja po rampi do vrednosti referentne frekvencije i prosleđuje se motoru
3. ***drive1_angle.full –*** tačna pozicija rotora motora u opsegu od 0 do 2π
4. ***drive1_param_scale_angle*** – parametar kojim se vrednost pozicije skalira u opseg od 0 do 2π

```
      ************************************************************************
      //  - Funkcija koja implementira U/f regulaciju
      //
      ************************************************************************
      void Drive1_UF( void )
      {
(1)         // Priprema nove frekvencije za U/f, referenca "ide po rampi"
            if ( drive1_state == ON )
            {
                  // drive1_uf_mscounter x 100us period expired
(2)               if ( ++drive1_uf_mscounter == 10 )
                  {
(3)                     if ( drive1_f_ref > drive1_f )
                        { drive1_f++; }
                        else if ( drive1_f_ref < drive1_f )
                        { drive1_f--; }
(4)
                        drive1_uf_mscounter = 0;
                  }
            }
            else
            {
                  drive1_angle.full = 0;
                  drive1_f = 0;
            }

(5)         drive1_angle.full += drive1_f * drive1_param_scale_angle;

            OVERFLOW_MODE_ENABLE();

            // Citanje sinusne tabele; odredjivanje sin(theta) i cos(theta)

            drive1_sine_table_index = (unsigned int) ( drive1_angle.part.high >>
      6 ) & 0x03FF;
            drive1_sin_theta       =       PFUNC_wordRead(       &sine_lut       +
      drive1_sine_table_index );
            drive1_sine_table_index  =  (  drive1_sine_table_index  +  256  )  &
      0x03FF;
            drive1_cos_theta       =       PFUNC_wordRead(       &sine_lut       +
      drive1_sine_table_index );
```

```
(6)         if ( drive1_f < drive1_param_f_nom )
            {
                  drive1_u = drive1_f * drive1_param_scale_u;
            }
            else if ( drive1_f < 0 )
            {
                  drive1_u = -0x7FFF;
            }
            else
            {
                  drive1_u = 0x7FFF;
            }

            calc_temp.full   =   drive1_u   *   drive1_cos_theta   +   drive1_u   *
      drive1_cos_theta;
(7)         drive1_ualpha = calc_temp.part.high;
            calc_temp.full   =   drive1_u   *   drive1_sin_theta   +   drive1_u   *
      drive1_sin_theta;
(8)         drive1_ubeta = calc_temp.part.high;
```

Slika 3.4 – S kod funkcije kojom se implementira U/f upravljanje

Zadaci i pitanja

1. Polazeći od podataka za dati motor, odrediti odnos napona i učestanosti koji se treba imati kod U/f napajanja motora. Kako se menja napon a kako učestanosti u režimu slabljenja polja?
2. Koju vrednost treba da ima drive1_f_ref da bi se motor obrtao brzinom bliskom 1000 o/min?
3. Koja je razlika između drive1_f_ref i drive1_f? Koje vreme je potrebno da se motor ubrza od nule do 1000 o/min?
4. Opisati ulogu i razmeru varijable drive1_angle.full. Koju vrednost i zašto treba da ima drive1_param_scale_angle?
5. Koji parametar određuje odnos U/f i kako se on određuje?
6. Na osnovu koda, nacrtati blok dijagram koji prikazuje tok signala počevši od komande za učestanost pa do modulacionih signala.
7. Zašto je potrebno aktivirati funkciju ***OVERFLOW_MODE_ENABLE()?***

4. Digitalna regulacija struje

Uputstvo se odnosi na deo laboratorijskih vežbi MOT09 (TEMPUS) u kome je predmet vežbanja digitalna regulacija struje u okviru električnih pogona sa asinhronim i sinhronim mašinama. Vežbe se odvijaju u laboratoriji 40a na Elektrotehničkom fakultetu. Pre dolaska na vežbu student treba da pročita ovo uputstvo kao i dokumente na koje se uputstvo poziva, a pre svega prilog koji obrađuje postupak unosa programskog koda u memoriju Digitalnog Signal Procesora (DSP) to jest „postupak programiranja".

Pitanja za proveru znanja nalaze se u poglavlju 4.3. dok se zadaci predviđeni za rad u laboratoriji nalaze u poglavljima 4.4 i 5.1. ovog dokumenta. Nakon rada u laboratoriji očekuje se da student popuni izveštaje rada u laboratoriji koji se nalazi na krajevima poglavlja 4 i 5. U izveštaj se unose odgovori na pitanja iz 4.3. kao i rezultati koji se dobijaju tokom rada u laboratoriji.

U poglavljima 4.1. i 4.2. nalaze se teorijska objašenjenja svih segmenata potrebnih za uspešno savladavanje laboratorijske vežbe.

Kroz ovo uputstvo student se upoznaje sa nekim od osnovnih alata u regulaciji struje, preko teorijskih osnova do krajnjeg kodiranja u programskom jeziku C. U uvodnom delu student treba da sagleda sledeće:

- potrebu za regulacijom struje
- način napajanja motora
- merenje i digitalizacija struje motora
- implementaciju regulatora struje na DSP platformi
- način upravljanja energetskim pretvaračem (invertorom) koji se koristi za napajanje motora

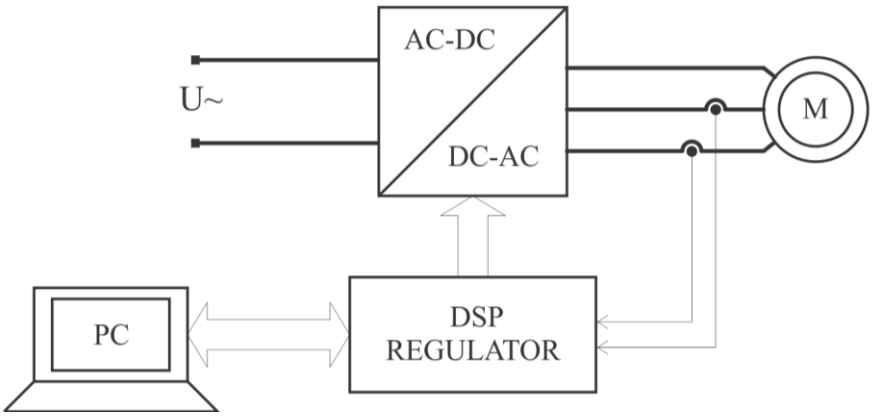
U narednih nekoliko poglavlja dat je teorijski uvod.

4.1. Uvod

U laboratorijskim vežbama se koriste motori za naizmeničnu struju. U okviru industrijskih i drugih primena električnih motora u upravljanju kretanjem, u regulaciji pozicije i brzine, neophodno je raspolagati mogućnošću da se zadaje pokretački momenat. Budući da je momenat određen fluksom i strujom mašine, brže izmene momenta zahtevaju brže izmene statorske struje, što se može postići digitalnom regulacijom struje. Brzina odziva celokupnog sistema za upravljanje kretanjem zavisi od karakteristika regulatora struje koji je predmet vežbanja. Dakle, poželjno je proučiti regulaciju struje motora jer većina aplikacija kontrole momenta motora uključuje regulaciju struje.

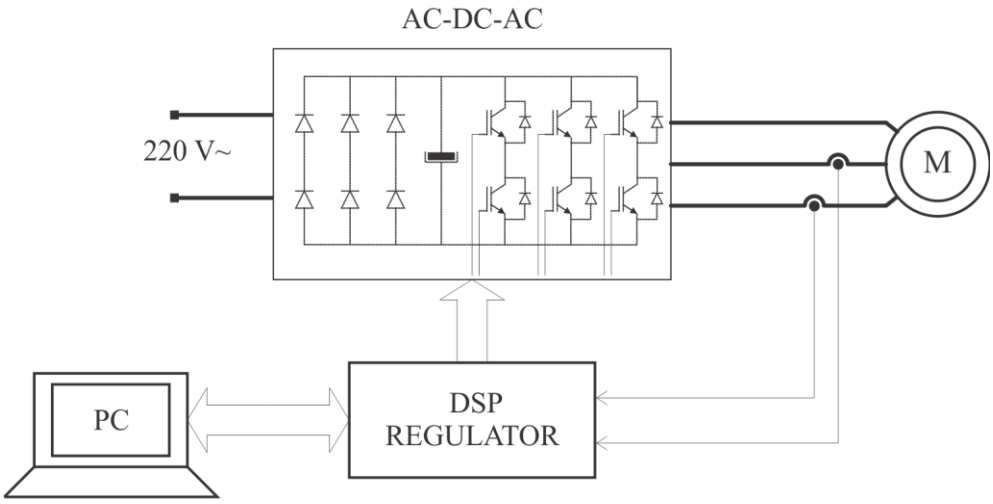
Digitalna regulacija struje podrazumeva obradu informacija o struji pogona u digitalnom (diskretnom) domenu na osnovu koje se generiše komanda i upravljanje pogona u cilju ostvarivanja željene struje. Neke od osnovnih prednosti digitalne regulacije struje u odnosu na analognu su brže izvršavanje algoritma, mogućnost realizacije znatno složenijih algoritama upravljanja, jednostavna zamena algoritma upravljanja na istom uređaju koja ne podrazumeva neke hardverske izmene a i sam hardware koji izvršava regulaciju je kompaktniji od analognog. Ono što je nedostatak digitalne regulacije u odnosu na analognu je neophodnost određenog sklopa koji omogućava pretvaranje signala koji su u analognom domenu u informacije u digitalnom domenu.

Napajanje motora se vrši preko energetskih pretvarača koji imaju za cilj da prilagode napone i struje na priključnim krajevima mašine tako da se ostvari željena funkcija pogona. Na Slici 4.1 prikazana je uprošćena blok šema jednog digitalno regulisanog pogona, korišćenog u laboratorijskoj vežbi.



Slika 4.1 Blok šema laboratosijske postavke

Na Slici 4.1 je moguće uočiti nekoliko celina. Napon gradske mreže se ispravljačem ispravlja i tako ispravljen se preko invertora dovodi na krajeve motora. Invertor omogućava napajanje motora naponom promenljive učestanosti i amplitude. Regulator na ulazu dobija informacije o struji motora i na osnovu toga generiše komandu invertoru u vidu impulsa koji kontrolišu rad prekidačkih elemenata. Informacija o struji motora se dobija sa davača struje koji daju informaciju o ostvarenoj struji u motoru. Takođe postoji mogućnost da PC računar bude povezan na regulator i da tako u realnom vremenu kontroliše rad regulatora što znatno širi spektar mogućnosti pogona. U laboratorijskim vežbama su kao ac-dc-ac pretvarači korišćeni industrijski invertori kompanije *MOOG*, algoritam je implementiran na Digitalnom Signal Procesoru (DSP) kompanije Texas Instruments, dok su motori kojima se upravlja motori za naizmeničnu struju (asinhroni i sinhroni). Na Slici 4.2 je prikazan nešto detaljniji sistem koji je predmet rada u ovoj laboratorijskoj vežbi.



Slika 4.2 Detaljnija blok šema laboratorijske postavke

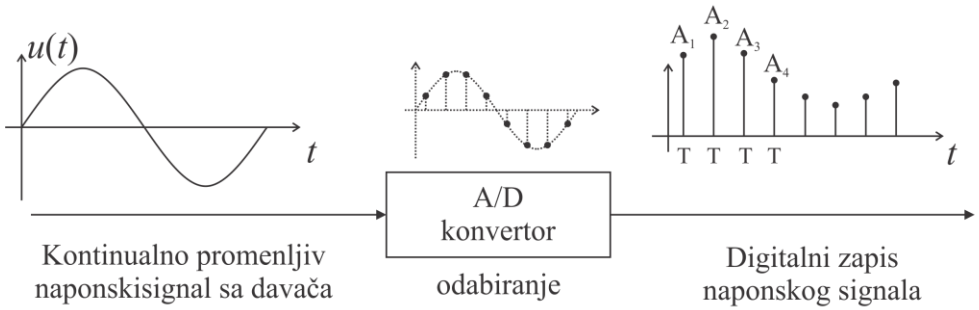
Detaljniji opis opreme koja se koristi u laboratorijskoj vežbi dat je u prilogu, pod nazivima Dodatak A i Dodatak B.

4.2. Merenje i upravljanje strujom

U narednom delu uputstva će biti predstavljeni redom svi delovi pogona prvenstveno sistem za merenje struje zatim struktura regulatora, način upravljanja invertorom Impulsno Širinskom Modulacijom (PWM) a na kraju će biti detaljnije prikazan hardware na kome se izvodi vežba. Kroz uputstvo će se na bitnim mestima prikazivati *C* kod kako bi studentu bio omogućen uvid u praktične aspekte implementacije regulacije struje. Priložen kod se nalazi u projektnim direktorijumima i dostupan je studentima na vežbama. O projektnim direktorijumima će biti reči u poglavljima 4.4. i 5.

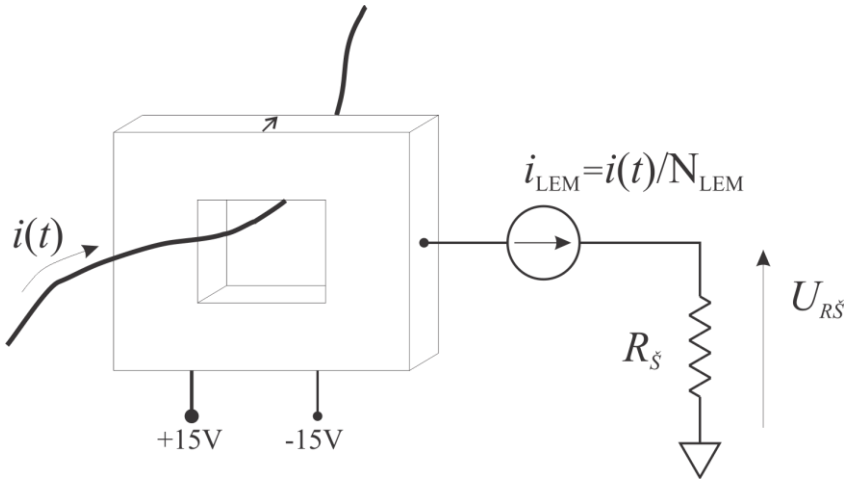
4.2.1. Merenje struje u digitalno regulisanom pogonu

Za merenje struje koriste se davači struje na bazi Hall-ovog efekta. Princip rada davača detaljnije se obrađuje na predavanjima iz predmeta Digitalno upravljanje pretvaračima i pogonima 1. Prikazan je kompletan sistem za merenje struje u pogonima sa digitalnom regulacijom gde će biti napravljen manji osvrt na davače struje, odabiranje signala kao i na određena ograničenja koja se moraju uvesti. Zadatak mernog sistema je da struju iz kontinualnog domena prevede u digitalni zapis, kao što prikazuje Slika 4.3.



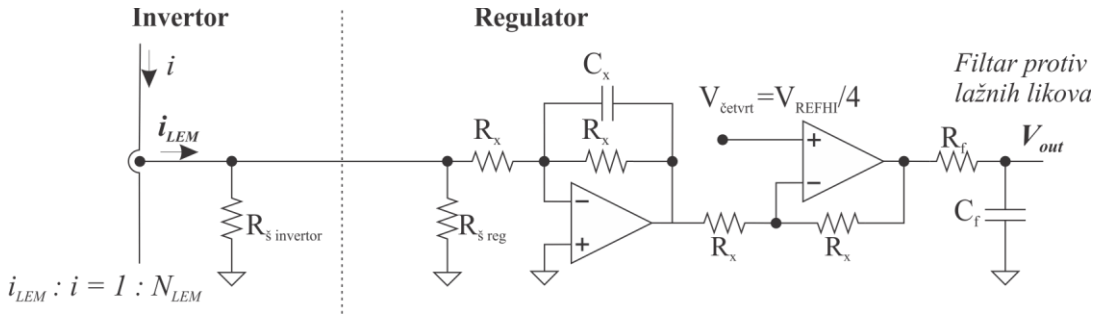
Slika 4.3 Princip digitalizacije signala

U motorima koji se koriste u vežbi u namotajima statora postoji naizmenična struja. Davači struje (u daljem tekstu LEM) na svom izlazu daju struju koja je proporcionalna merenoj struji. Princip rada LEM davača prikazan je na Slici 4.4.



Slika 4.4 Davač struje

Struja koja se ima na izlazu iz davača je N_{LEM} (na primer 1000) puta manja od merene struje. U slučaju da je struja naizmenična (prostoperiodična) amplitude 1 A, struja i_{LEM} je takođe prostoperiodična, amplitude 1 mA (amplitude manje N_{LEM} puta od merene). Za dalju obradu (odabiranje) potrebno je imati naponski signal, a ne strujni. Zato se struja i_{LEM} usmerava na otpornik $R_Š$. U okviru DSP-a postoji A/D periferijski uređaj koji pretvara analogne napone iz opsega 0 do 3,3 V u digitalni zapis (broj). U zavisnosti od vrednosti ulaznog napona, rezultat koji se ima nakon pretvaranja, menja se u određenom opsegu (kasnije je objašnjeno koji je to opseg). Zato je signal napona na elementu $R_Š$, koji se menja od $\pm R_Š\, i_{LEM}$ neophodno obraditi tako da se menja u opsegu 0 ÷ 3,3 V. Ako se otpornost šanta odredi kao 3,3 V / 2 $I_{LEM(max)}$ a potom se na napon šanta nadoda 1.65 V, postiže se da pri struji koja je jednaka nuli A/D konvertor vrši konverziju napona od 1,65 V. Tako se pri maksimalnoj struji konvertuje napon od 3,3 V dok se pri minimalnoj struji konvertuje napon od 0 V. U tu svrhu je implementirana neka vrsta analognog filtra koji obavlja prethodno opisanu funkciju. Filter je detaljno predstavljen Slikom 4.5.

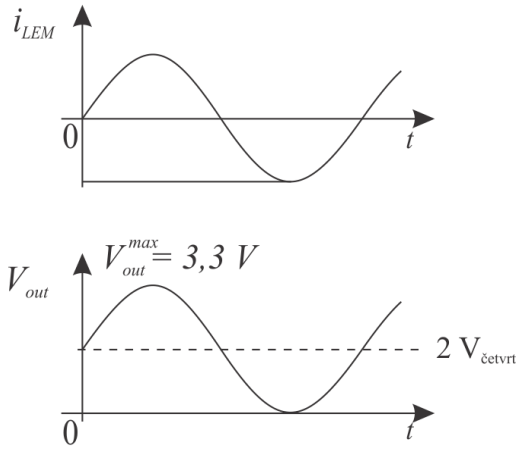


Slika 4.5 Šema analognog filtriranja signala

Sa leve strane Slike 4.5, u delu koji ukazuje na invertor, odvojeno tačkastom linijom, nalaze se davač struje i šant koji su locirani unutar invertora. Sa desne strane u delu označenom kao regulator, prikazan je sklop koji se nalazi na DSP ploči i koji zapravo sabira napon dobijen na $R_Š$ (koji ***mora*** biti u opsegu ±1,65 V) sa 1,65 V. Tako se dobija napon V_{out} koji se menja u granicama 0 ÷ 3,3 V i koji može da se vodi na AD konvertor. Na Slici 4.6 prikazan je signal LEM-a i signal koji se ima nakon filtracije.

Sledeći izraz pokazuje kako se menja izlazni napon V_{out} u zavisnosti od struje faze *i*.

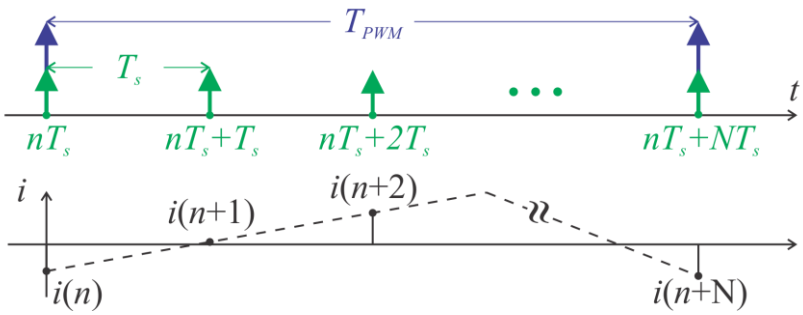
$$V_{out} = f(i) = 2\mathrm{V}_{\text{četvrt}} + i_{LEM}\mathrm{R}_{\text{Še}} = 2\mathrm{V}_{\text{četvrt}} + i / \mathrm{N}_{LEM}\mathrm{R}_{\text{Še}}$$



Slika 4.6 Zavisnost filtriranog napona od struje davača

AD konvertor je 10-bitni, dok je sabirnica podataka 16-bitna. Na korišćenom procesoru rezultat konverzije se smešta u registar dužine 16 bita i to tako da se koristi samo 10 bita, bitovi od 6 do 15. Tako dobijen rezultat se kreće u granicama 0 – $FFC0_{hex}$. Broj $FFC0_{hex}$ odgovara maksimalnom naponu na $R_Š$ a samim tim i maksimalnoj struji dok broj 0_{hex} odgovara minimalnom naponu na $R_Š$ a samim tim i minimalnoj struji, onoj pri kojoj se ima napon na šantu u iznosu od -1,65 V. U trenutku kada je merena struja jednaka nuli u AD konvertoru se kao rezultat konverzije dobija broj $7FC0_{hex}$. Nakon konverzije, vrši se pomeranje broja za 6 mesta, koji iz oblika NNNN NNNN NNXX XXXX prelazi u oblik 0000 00NN NNNN NNNN.

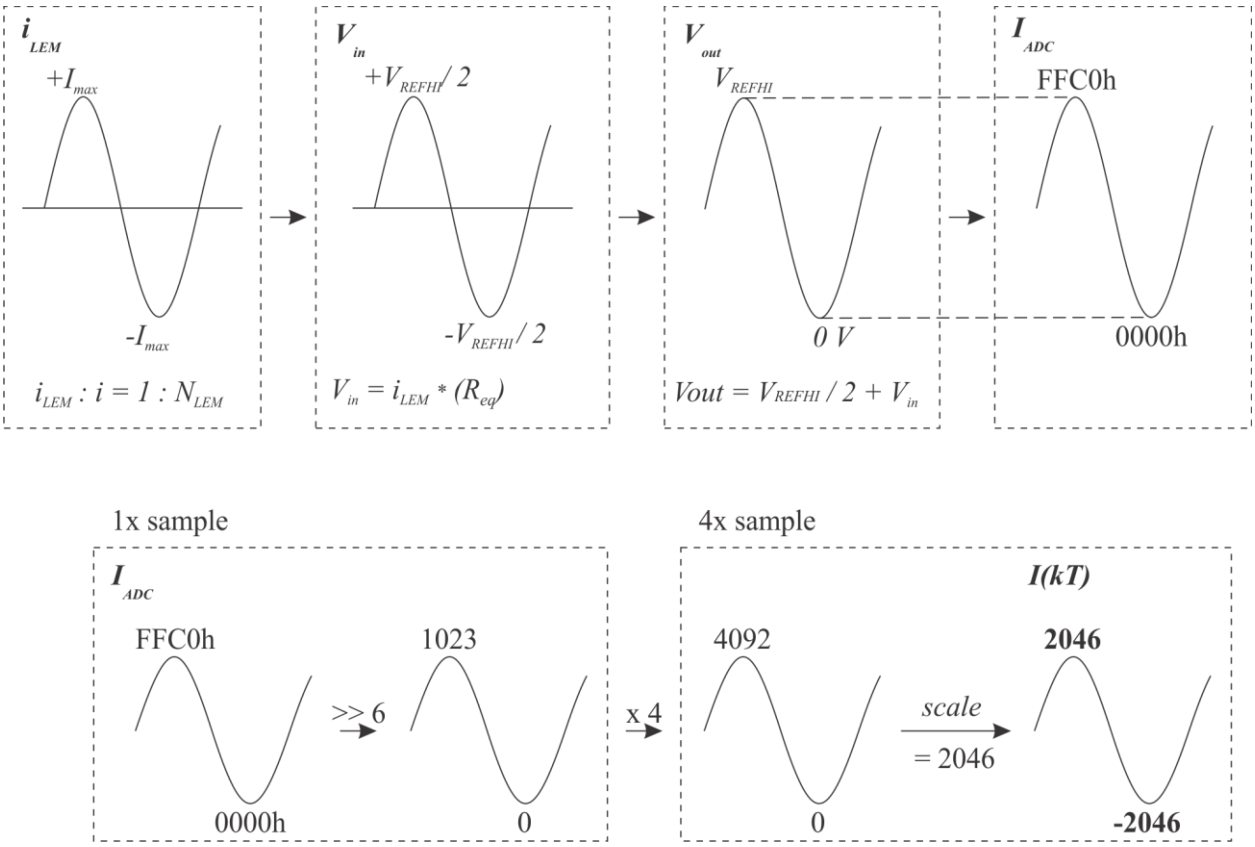
Izlazni filtar u sklopu datom na Slici 4.5 je filtar protiv lažnih likova kao i filtar koji ima zadatak da filtrira šum. Šum u digitalno regulisanim pogonima je nepoželjan ali i neizbežan. Kao dodatna tehnika za umanjivanje šuma iz sistema merenja struje primenjena je *oversampling* tehnika, o čemu se detaljnije govori na predavanjima iz predmeta Digitalno upravljanje pretvaračima i pogonima 1.



Slika 4.7 Princip oversampling-a

Oversempling podrazumeva uzimanje više odbiraka u jednom periodu izvršavanja algoritma, kao što je dato na Slici 4.7. U ovom slučaju je realizovana tako da se uzima 4 odbirka struje u jednom periodu. Pošto se smatra da je srednja snaga elektromagnetskog šuma jednaka nuli, sabiranje korisnog signala superponiranog sa šumom i to 4 puta u periodu izvršavanja algoritma dovodi do smanjivanja negativnih efekata elektromagnetskog šuma na merenu veličinu.

Na Slici 4.8 je slikovito objašnjen proces transformacije struje u konačni binarni broj.



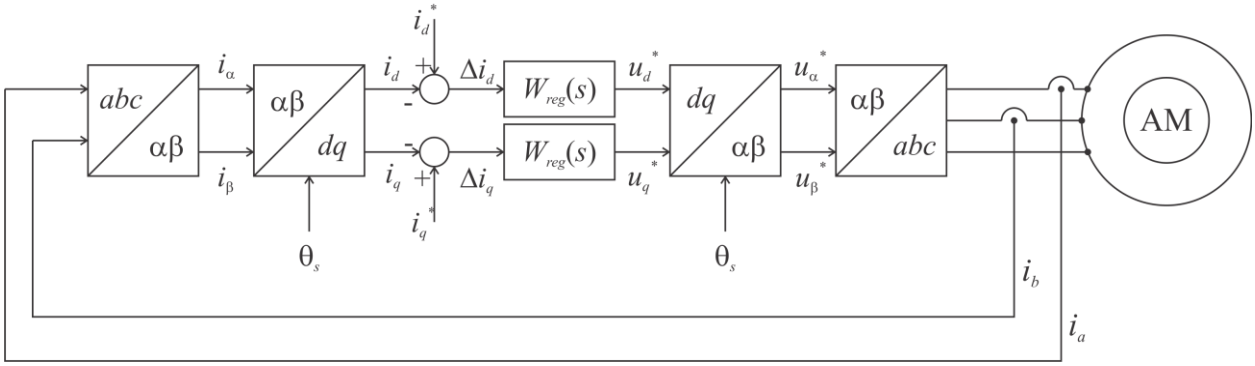
Slika 4.8 Razmere struje

Pošto se struja odabira 4 puta u jednom periodu izvršavanja algoritma i nadodaje na prethodno odabranu vrednost, merena struja se kreće u opsegu od 0_{hex} do $0FFF_{hex}$. Dakle na segmentima Slike 4.8 je prikazan slučaj punog opsega struje.

4.2.2. Struktura regulatora struje

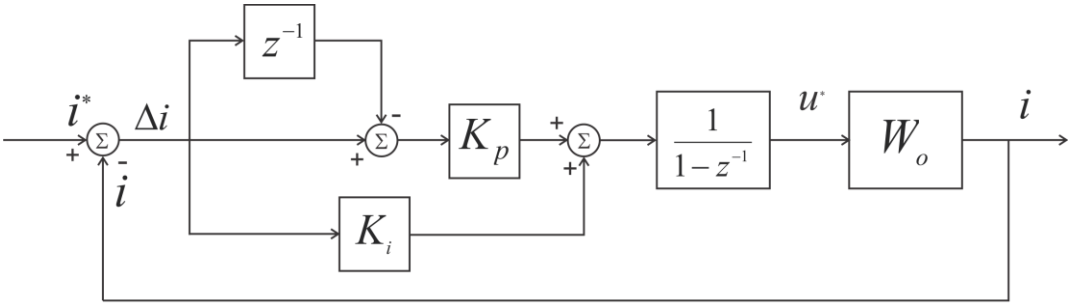
Regulator struje je izveden u sinhrono rotirajućem koordinatnom sistemu (na dalje će biti korišćen termin *dq* sistem). Ulaz u regulator struje je referentna struja dok je izlaz referentni napon. Referentna struja se dobija kao izlaz iz regulatora više hijerarhije (na primer regulator brzine) dok se referentni napon, nakon dalje obrade (*dq – abc* transformacija i *PWM* modulacija) preko invertora dovodi na faze motora. Detaljnije informacije i objašnjenja su dati na predavanjima i u literaturi pa u ovom uputstvu neće biti više reči o tome.

Tok izvršavanja algoritma prikazan je na Slici 4.9.



Slika 4.9 Kompletna struktura regulatora struje

U pogonu se mere 2 struje, i_a i i_b. Primenom obrtnih transformacija - transformacija stanja na merene struje, dobijaju se struje i_d i i_q. Regulacija struje na dalje se vrši nad dobijenim strujama u *dq* sistemu. Blok šema regulatora data je na Slici 4.10.



Slika 4.10 Regulator struje u inkrementalnoj formi

Regulator je izveden zasebno za *d* i *q* osu po principu koji je dat na Slici 4.10. Neophodno je uočiti da regulator struje nema dislocirano proporcionalno dejstvo u lokalnu granu kao ni unakrsno dejstvo (unakrsno delovanje *d* ose na *q* i obrnuto). O prednostima regulatora koji ima proporcionalno dejstvo izmešteno u lokalnu granu govori se na kursu Digitalno upravljanje energetskim pretvaračima i pogonima 2 pa u ovom uputstvu neće biti obrađivan.

Deo koda koji u okviru regulatora struje vrši transformaciju koordinata stanja (*abc* - *dq*) predstavljen je u sledećih nekoliko linija (kompletan kod se nalazi u projektnim direktorijumima koji se koriste za vežbe o čijim lokacijama će biti reči kasnije - Tabela 4.1, Poglavlje 4.4.).

Napomena: u kodu se pojavljuje promenljiva *calc_temp* i nju treba smatrati kao 32-bitni broj. Cilj je da se jednom promenljivom može pristupati 32-bitnim podacima iako je sabirnica podataka 16-bitna. Potreba za pristupanje 32-bitnim brojevima je očigledna imajući u vidu da je tolika dužina broja koji je rezultat množenja dva 16-bitna broja.

```
// Ia, Ib, Ic -> Ialpha, Ibeta; Same peak values remain; K = 2/3;
// | i_alpha | = | 1            0              | | Ia |
// | i_beta  | = | sqrt( 3 ) / 3  2 * sqrt( 3 ) / 3 | | Ib |
drive1_ialpha = drive1_ia;
calc_int = 18918; // sqrt(3) / 3 * 0x7FFF
calc_temp.full =          //calc_temp je32-o bitna struktura i sadrzi gornjih 16 i donjih 16 bita
        calc_int * drive1_ia
        + calc_int * drive1_ib
        + calc_int * drive1_ib;
drive1_ibeta = calc_temp.part.high << 1;
// Racunanje sin( theta_dq ), cos( theta_dq )
drive1_theta_s_el = ( drive1_theta_s_el >> 6 ) & 0x03FF;
drive1_sin_theta = PFUNC_wordRead( &sine_lut + drive1_theta_s_el );
drive1_theta_s_el = ( drive1_theta_s_el + 256 ) & 0x03FF;
```

```
drive1_cos_theta = PFUNC_wordRead( &sine_lut + drive1_theta_s_el );
// Ialpha, Ibeta -> Id, Iq;
// | i_d | = | cos( theta )   sin( theta )  | | Ialpha |
// | i_q | = | -sin( theta )   cos( theta ) | | Ibeta  |
calc_temp.full =  drive1_cos_theta * drive1_ialpha + drive1_sin_theta * drive1_ibeta;
drive1_id = calc_temp.part.high << 1;
calc_temp.full = -drive1_sin_theta * drive1_ialpha + drive1_cos_theta * drive1_ibeta;
drive1_iq = calc_temp.part.high << 1;
```

U kodu koji je predstavljen, koji realizuje dq transformaciju, treba uočiti da se ne pominje struja faze *c*. Razlog tome je merenje samo dve struje na pogonu što se pokazuje kao dovoljno u regulaciji pogona. Kako postoje 3 fazne struje samo 2 su nezavisno promenljive, struja faze c se može dobiti na osnovu merenih struja faze a i b ($i_c = -i_a - i_b$).

Kod koji izvršava regulaciju struje u skladu sa prethodno navedenom blok šemom regulatora je naveden u narednih nekoliko linija teksta.

```
calc_temp.full = ( drive1_id_ref << 16 ) - ( drive1_id << 16 );
drive1_id_error = calc_temp.part.high;    // racunanje greske struje d
calc_temp.full -= (long) drive1_id_error_old << 16;
drive1_id_error_inc = calc_temp.part.high;        // racunanje inkrementa greske (potrebno zbog
                                // regulatora inkrementalnog realizovanog u inkrementalnoj formi
drive1_id_error_old = drive1_id_error;

calc_temp.full = ( drive1_iq_ref << 16 ) - ( drive1_iq << 16 );
drive1_iq_error = calc_temp.part.high;   // racunanje greske struje q
calc_temp.full -= (long) drive1_iq_error_old << 16;
drive1_iq_error_inc = calc_temp.part.high; // racunanje inkrementa greske (potrebno zbog
                                // regulatora inkrementalnog realizovanog u inkrementalnoj formi
drive1_iq_error_old = drive1_iq_error;
// PI regulator struje u d osi
// d_ud = kp * d_e_id + ki * e_id
calc_temp.full =
        drive1_param_iloop_kp * drive1_id_error_inc
        + drive1_param_iloop_ki * drive1_id_error;
// Neophodna normalizacija: d_ud * 2 ^ 10
// Mora da se radi sabiranje zbog prekoracenja (overflow)
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
calc_temp.full += calc_temp.full;
// ud += d_ud
drive1_udloop.full += calc_temp.full;
```

Potrebno je napomenuti da je u prethodnom kodu primenjen račun sa 32-bitnom tačnošću. Kako je arhitektura procesora zasnovana na 16-bitnim sabirnicama podataka, korišćeni su tipovi podataka *struct* i *union*, što neće biti predmet podučavanja pa i neće biti dalje obrađivano.

Kao što je već pomenuto, prikazana blok šema regulatora i programski kod primenjuju se posebno za *d* i *q* osu. Tako i prikazan postupak dobijanja naponske reference za *d* osu identično se primenjuje i za *q* osu..

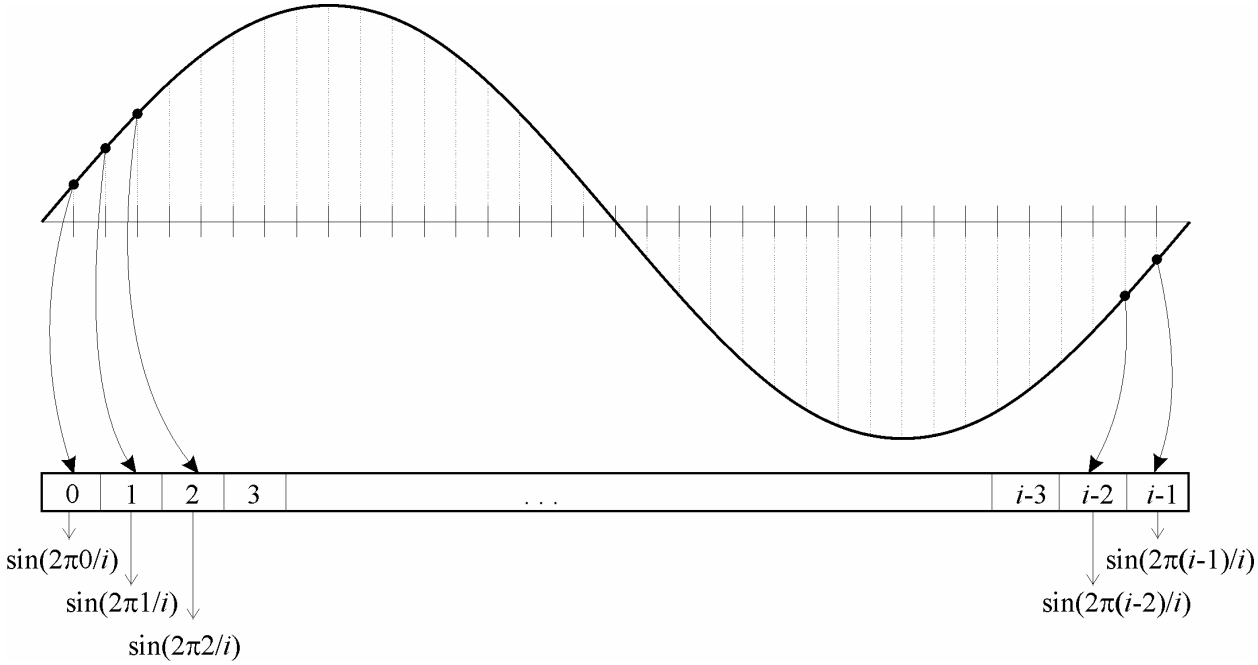
4.2.3. Računanje trigonometrijskih funkcija

Za realizaciju abc-dq transformacije potrebno je poznavanje određenih trigonometrijskih funkcija položaja – ugla θ_{dq}. O tome kako se dobija taj ugao govori se u laboratorijskim vežbama „Indirektno vektorsko upravljanje“ (poglavlje 6.11.), dok će ovde biti predstavljen postupak računanja trigonometrijskih funkcija u funkciji dobijenog ugla.

U programskom jeziku *C* postoje biblioteke koje omogućavaju računanje trigonometrijskih funkcija. Međutim, njihove vrednosti se dobijaju numeričkim metodama, razvijanjem u red argumenta funkcije i ispostavlja se da bi na ovaj način procesor utrošio isuviše mnogo vremena za računanje.

Kako je i predstavljeno u odeljku 4.2.2 gde je prikazan kod koji realizuje *dq* transformaciju u okviru regulacije struje (// Racunanje sin(theta_dq), cos(theta_dq)), odgovarajuće vrednosti sinusne funkcije se mogu pročitati iz unapred pripremljenih tabela (*look-up table*) funkcijom PFUNC_wordRead. To praktično znači da je formiran jedan niz određene dužine u koji se smeštaju vrednosti sinusne funkcije. Svaki član niz ima onu vrednost koja odgovara sinusnoj funkciji ugla koji odgovara poziciji tog člana u nizu.

U laboratorijskim vežbama postoji tabela sa vrednostima sinusne funkcije od 1024 člana i zapisana je u označenom formatu sa 16 bita. Na Slici 4.11 prikazan je princip formiranja niza koji sadrži vrednosti sinusne funkcije. Na slici je dužina niza predstavljena sa *i* što u realnom slučaju na laboratorijskim postavkama predstavlja 1024. Pošto su brojevi zapisani u 16-bitnom brojnom sistemu maksimalna vrednost sinusoide je zapisana brojem $7FFF_{hex}$ dok je minimalna vrednost zapisana sa 8000_{hex}. Za rad u laboratoriji nije neophodno detaljno poznavanje formiranje sinusne tabele pa su ovde predstavljeni samo osnovni aspekti rada sa tabelama (*look-up table*).



Slika 4.11 Sinusna tabela

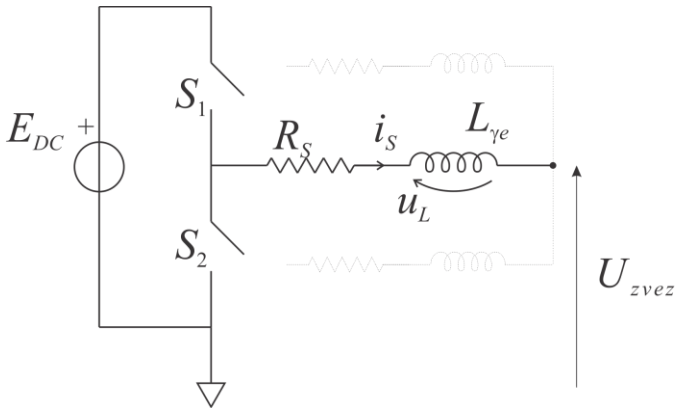
4.2.4. PWM modulacija

Napajanje digitalno kontrolisanim motorima naizmenične struje vrši se pomoću pretvarača snage, koji koriste poluprovodničke prekidačke komponente. Na faze motora potrebno je dovoditi napon varijabilne amplitude i frekvencije. Priroda mehaničkog podsistema je takva da na dinamiku rada motora ne utiču visokofrekventne pojave u električnom podsistemu. Promenu srednje vrednosti napona je moguće ostvariti prekidanjem nekog napona približno konstantne vrednosti (E_{DC}). Ukoliko se to radi veoma često (10 000 puta u sekundi) postiže se brza promena srednje vrednosti napona. Ako se na svakih 100 μs zadaje različita srednja vrednost napona onda je moguće kontrolisanje amplitude kao i učestanosti naizmenične komponente napona koji se dovodi na faze motora. U ovom odeljku će biti prikazani neki osnovni aspekti primene Impulsno Širinske Modulacije (*eng. Pulse Width Modulation*).

Na Slici 4.12 prikazan je princip napajanja jedne faze motora putem invertora. Prekidači S_1 i S_2 omogućuju dovođenje napona vrednosti E_{DC} ili nula na fazu motora. Vremenski interval u kome je uključen gornji prekidač se obeležava sa t_{on}. U jednoj periodi T rada ovakvog sklopa srednja vrednost dovedenog napona na fazu motora je jednaka

$$U_{sr} = \frac{t_{on}}{T} E_{DC}$$

što je jedan od izraza koji je pominjan na predavanjima iz predmeta Digitalno upravljanje pretvaračima i pogonima 1.



Slika 4.12 Jedna faza motora napajana energetskim pretvaračem

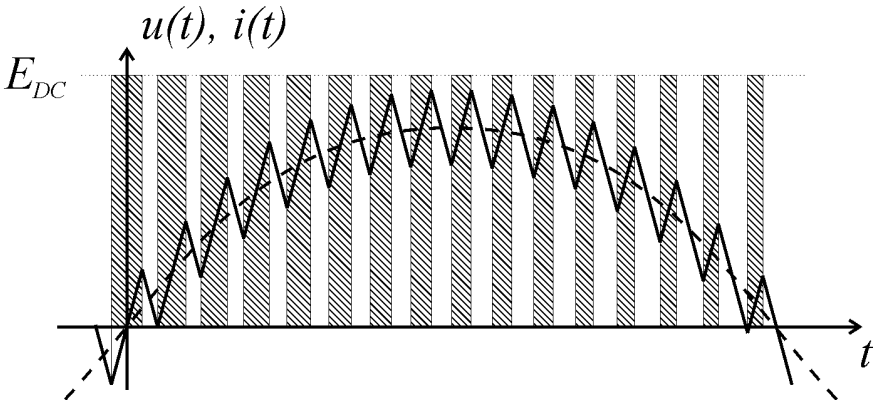
Srednja vrednost napona u jednom intervalu T srazmerna je šrafiranoj površini na Slici 4.13. Kako bi se od ovakvog četvrtastog napona koji se dovodi na fazu motora napravio sinusoidalni napon očigledno je da je potrebno varirati vreme t_{on} tako da se zadovolji sledeće

$$u_a = \frac{E_{DC}}{2} + A\frac{E_{DC}}{2}\sin(\omega_s t)$$
$$u_b = \frac{E_{DC}}{2} + A\frac{E_{DC}}{2}\sin(\omega_s t - \frac{2\pi}{3})$$
$$u_c = \frac{E_{DC}}{2} + A\frac{E_{DC}}{2}\sin(\omega_s t + \frac{2\pi}{3})$$

gde A predstavlja amplitudu napona koji se dovodi i može uzeti vrednost [0..1]. Da bi se ostvario ovaj uslov vreme t_{on} treba da se menja na sledeći način

$$t_{on}^a = \frac{T_{PWM}}{2} + A\frac{T_{PWM}}{2}\sin(\omega_s t)$$
$$t_{on}^b = \frac{T_{PWM}}{2} + A\frac{T_{PWM}}{2}\sin(\omega_s t - \frac{2\pi}{3})$$
$$t_{on}^c = \frac{T_{PWM}}{2} + A\frac{T_{PWM}}{2}\sin(\omega_s t + \frac{2\pi}{3})$$

Ukoliko se ispoštuju prethodne pretpostavke može se reći da se naizmenične komponente napona na fazama motora menjaju po sinusnom zakonu pa se tako ima i zavisnost struje koja je prikazana na Slici 4.13 uz pretpostavku da je opterećenje induktivno.



Slika 4.13 Zavisnost struje od napona na izlazu PWM jedinice

Bitno je uočiti da se variranjem t_{on} (smanjivanjem) dobija promena struje kao na slici, za vreme trajanja t_{on} (šrafirane površine) struja raste dok u ostalim vremenskim intervalima struja opada.

Zadatak regulatora struje je zapravo računanje vremena t_{on} za sve 3 faze motora i nakon toga se, u sledećoj periodi rada algoritma, u skladu sa izračunatim vremenima pale odnosno gase tranzistori u invertoru.

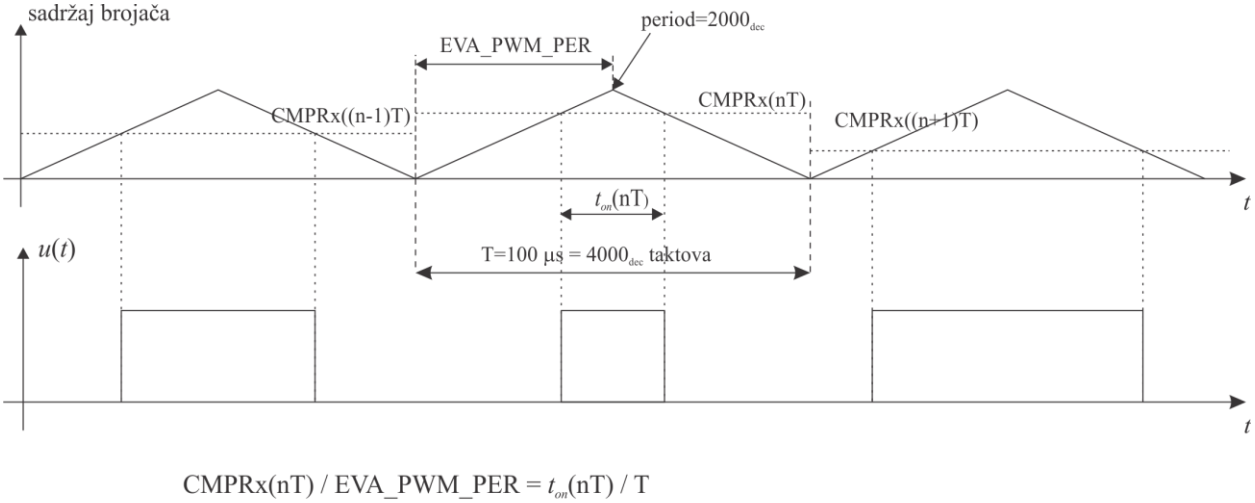
Treba napomenuti da se kao izlaz regulatora struje dobijaju referentne vrednost napona u_d i u_q u formi 16-bitnog označenog broja i da se na njih mora primeniti inverzna transformacija stanja, iz *dq* sistema u *abc*. Na osnovu dobijenih trenutnih referentnih vrednosti za napone sve 3 faze računaju se vremena t_{on} i upisuju se u određene registre PWM jedinice u formi broja. Sekvenca koda koja izvršava ovu funkciju u laboratorijskoj vežbi je data u nastavku.

```
// Ud, Uq -> Ualpha, Ubeta
// | u_alpha | = | cos( theta)      -sin( theta)  | | u_d |
// | u_beta  | = | sin(theta)       cos( theta ) | | u_q |
calc_temp.full =
        drive1_cos_theta * drive1_ud + drive1_cos_theta * drive1_ud
        - drive1_sin_theta * drive1_uq - drive1_sin_theta * drive1_uq;
drive1_ualpha = calc_temp.part.high;
calc_temp.full =
        drive1_sin_theta * drive1_ud + drive1_sin_theta * drive1_ud
        + drive1_cos_theta * drive1_uq + drive1_cos_theta * drive1_uq;
drive1_ubeta = calc_temp.part.high;
// Ualpha, Ubeta -> Ua, Ub, Uc
// | Ua | = | 1     0            | | Ualpha |
// | Ub | = | -0.5  sqrt( 3 ) / 2  | | Ubeta  |
// | Uc | = | -0.5  -sqrt( 3 ) / 2 |
drive1_ua = drive1_ualpha;
calc_int = 28378;          // sqrt( 3 ) / 2 * 7FFF;
calc_temp.full = (long) drive1_ualpha << 15;
calc_temp2.full =
        calc_int * drive1_ubeta + calc_int * drive1_ubeta
        - calc_temp.full;
drive1_ub = calc_temp2.part.high;
calc_int = -28378;         // -sqrt( 3 ) / 2 * 7FFF;
calc_temp2.full =
        calc_int * drive1_ubeta + calc_int * drive1_ubeta
        - calc_temp.full;
drive1_uc = calc_temp2.part.high;
*****************************************************************************
        //ova funkcija se koristi nakon racunanja referentnih napona
        //upisivanje vrednosti u PWM compare registre
        EVA_LoadPWM( drive1_ua, drive1_ub, drive1_uc )
```

```
**************************************************************************
        //definicija funkcije za upisivanje vrednosti u PWM compare registre
#define EVA_LoadPWM( vola, volb, volc ) \
CMPR1 = (unsigned int) ( ( EVA_PWM_PER * -vola + ( (long) EVA_PWM_PER << 15 ) ) >> 16 ); \
CMPR2 = (unsigned int) ( ( EVA_PWM_PER * -volb + ( (long) EVA_PWM_PER << 15 ) ) >> 16 ); \
CMPR3 = (unsigned int) ( ( EVA_PWM_PER * -volc + ( (long) EVA_PWM_PER << 15 ) ) >> 16 );
        //kompletan kod u jeyiku C se nalazi u projektnim direktorijumima
```

U narednih nekoliko pasusa će ukratko biti pojašnjena funkcija koja osvežava *compare* (CMPR) registre PWM jedinice na početku izvršavanja algoritma (jednom u 100 μs).

Oscilator koji se koristi na DSP pločici obezbeđuje takt od 40 MHz. Glavna prekidna rutina se poziva na početku svakog ciklusa brojanja određenog brojača (na svakih 100 μs). U prekidnoj rutini se računa t_{on} što predstavlja „koeficijent ispune“ PWM-a (duty cycle). Brojač koji inicira prekidnu rutinu menja svoje stanje 40 000 000 puta u sekundi što je u vezi sa frekvencijom oscilatora. Brojač je iz određenih razloga organizovan tako da broji *up-down* što zapravo znači da bi brojač navršio ciklus brojanja mora da se inkrementira do neke unapred definisane vrednosti (EVA_PWM_PER) a onda počinje da se dekrementira do nule, kada ističe vreme od 100 μs. To znači da vreme koje je potrebno da brojač izbroji do vrednosti EVA_PWM_PER predstavlja polovinu vremena od 100 μs.



$$CMPRx(nT) / EVA\_PWM\_PER = t_{on}(nT) / T$$

Slika 4.14 Zavisnost PWM napona od sadržaja CMPR registra

Odavde se zaključuje da je vrednost EVA_PWM_PER jednaka 2000. Kada se vrednost brojača u fazi inkrementiranja izjednači sa vrednošću upisanom u određeni CMPR registar šalje se signal za uključivanje tranzistora, dok u fazi dekrementiranja brojača, ovaj događaj prouzrokuje isključivanje tranzistora. **Očigledno je da je vreme t_{on} u odnosu na vreme T u istom odnosu kao i broj upisan u CMPR registar u odnosu na period brojača EVA_PWM_PER**.

Kako se izračunati napon kreće u opsegu $\pm 7FFF_{hex}$ tako se za maksimalnu vrednost napona ima minimalna vrednost CMPR registara dok se za minimalni napon ima maksimalna vrednost CMPR registra. Vrednost koja treba da se upiše u CMPR registar po navedenoj logici se računa po sledećem principu

$$CMPR_A = \frac{\text{EVA\_PWM\_PER}}{2} \mp \frac{\text{EVA\_PWM\_PER}}{2} V_a$$

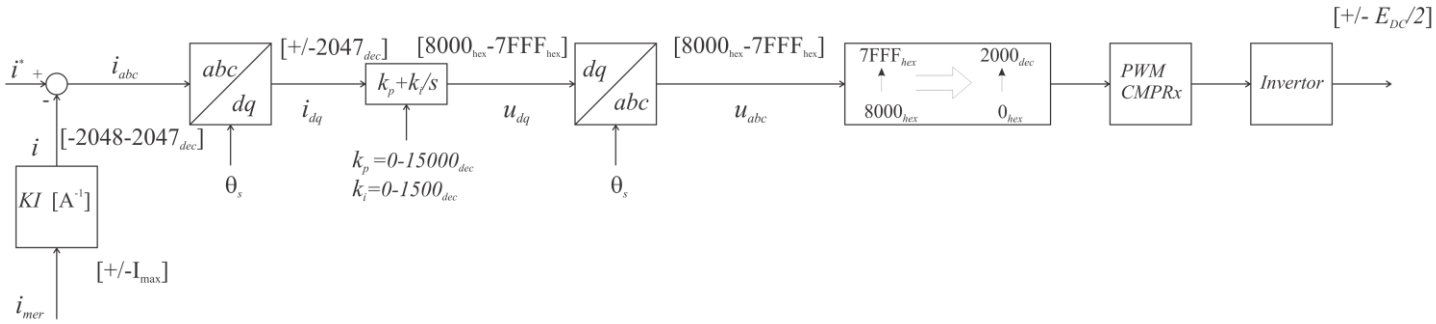
gde oznaka $CMPR_A$ predstavlja *compare* registar koji upravlja tranzistorima faze *a* dok V_a predstavlja svedenu vrednost izračunatog referentnog napona faze *a* i može uzimati vrednosti od maksimalne negativne do maksimalne pozitivne (8000_{hex} do $7FFF_{hex}$).

PWM Rezime: Srednja vrednost referentnog napona se računa na svakih 100 μs i takav napon, modulisan PWM modulacijom se dovodi na motor. PWM modulacija se praktično realizuje korišćenjem jednog od brojača koji postoje u DSP-u tako što se vrednost brojača poredi sa vrednošću koja na neki način predstavlja t_{on} u digitalnom zapisu. Kada vrednost brojača postane veća od unapred izračunatog vremena t_{on} šalje se signal za paljenje/gašenje određenog tranzistora.

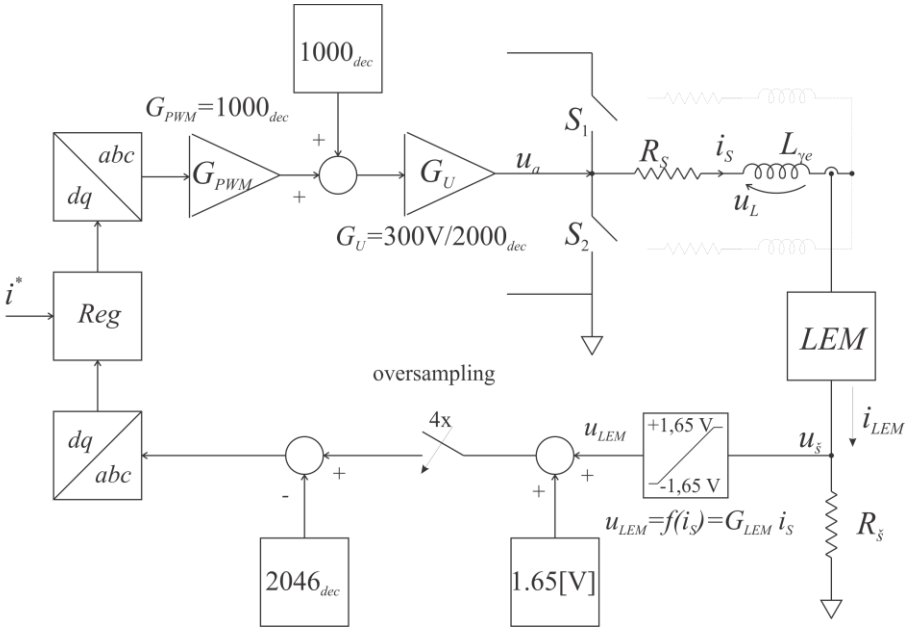
Digitalna regulacija struje - Rezime: Regulator struje, kao blok, ima svoje ulaze i izlaze. Regulator je izveden u *dq* koordinatnom sistemu i primenjuje se zasebno na *d* i *q* osu. Potrebno je izvršiti *abc*/*dq* transformaciju na osnovu merenih struja faza *a* i *b* koje se dobijaju kao izlaz iz LEM davača struje. Nakon toga izvršava se regulacija (u inkrementalnoj formi) i dobijaju se referentne vrednosti *d* i *q* napona. Nad dobijenim naponima se izvodi *dq*/*abc* transformacija i dalje se određuje broj koji se upisuje u CMPR registar PWM jedinice. Vrednost broja koji se upisuje u CMPR registar mora da se kreće u određenim granicama tako da dobijeni napon koji daje regulator mora da se skalira pre upisa u CMPR.

4.2.5 Razmere brojeva i tok informacija u regulatoru struje

Tok informacija sa razmerama brojeva koji se koriste se može videti na Slici 4.15. Predstavljen je regulator sa svojim ulazima i izlazima. Kao krajnji rezultat rada regulatora struje ima se broj ($0\text{-}2000_{dec}$) koji treba da se upiše u CMPR registre PWM jedinice. Kao što je već rečeno, broj upisan u CMPR registar predstavlja zapravo srednju vrednost napona koji se dovodi na faze motora.



Slika 4.15 Razmere u algoritmu



Slika 4.16 Blok šema implementiranog sistema

Na Slici 4.16 je prikazan blok dijagram regulatora struje. Određeni su i koeficijenti skale za neke blokove. Jedan od zadataka u okviru laboratorijskih vežbi (MOT09-4, Poglavlje 4.4) biće određivanje koeficijenta G_{LEM}.

4.3. Pitanja za proveru znanja

Pitanja za proveru znanja imaju za cilj da pomognu prilikom rekapituliranja gradiva neposredno pred početak praktičnog rada u laboratoriji. Odgovore na pitanja uneti u izveštaj koji se nalazi na kraju laboratorijske vežbe MOT09-4 „Podešavanje parametara regulatora struje“ (poglavlje 4.5.).

4.3.1. (Sinusna look-up tabela) Koji broj treba da se upiše u memoriju DSP-a a da predstavlja sinus od 60º?

4.3.2. (PWM) Ukoliko se algoritam izvršava na 15 kHz umesto na 10 kHz, koja vrednost treba da se upiše u EVA_PWM_PER?

4.3.3. (PWM) Ukoliko oscilator koji daje takt procesoru obezbeđuje 60 MHz umesto 40 MHz i ukoliko se želi izvršavanje algoritma na 100 μs, koliko iznosi vrednost promenljive EVA_PWM_PER?

4.3.4. (Merenje struje) Za struju od i_d od 0,7 A, koji je maksimalni broj zapisan u memoriju DSP-a a koja predstavlja vrednost struje i_q? (na pitanje treba odgovoriti u skladu sa izabranom postavkom koja se radi na vežbama i treba uzeti u obzir limit struje)

4.3.5. (Merenje struje) U kom opsegu se će se kretati rezultat A/D konverzije u slučaju da u motoru postoji struja koja je 2 puta veća od nominalne? (na pitanje treba odgovoriti u skladu sa izabranom postavkom koja se radi na vežbama).

4.4. Laboratorijska vežba MOT09 4 – Merenje struje korišćenjem davača struje

Vežba „Merenje struje korišćenjem davača struje“ spada u grupu vežbi iz regulacije struje. Struktura regulatora i sistem merenja struje dati su u poglavljima 4.1. i 4.2. uputstva „Digitalna regulacija struje“. Kao priprema za vežbu potrebno je odgovoriti na pitanja iz poglavlja 4.3 i dobijene odgovore uneti u izveštaj koji se nalazi u poglavlju 4.5 ovog dokumenta. U slučaju da student želi detaljnije da prouči opremu koja se koristi u vežbi, više može pročitati u Dodatku A i Dodatku B.

Cilj vežbe: sticanje praktičnog iskustva u

- merenju struje statora pomoću senzora na bazi Hall-ovog efekta (Slika 4.4)
- filtriranju analognih signala pre digitalizacije (Slika 4.5, poglavlje 4.2.1.)
- odabiranju analognih signala A/D konvertorom
- filtriranju digitalnih signala
- skaliranju unutrašnjih DSP promenljivih koje predstavljaju struju u određenoj razmeri

Fajlovi potrebni za vežbu nalaze se u odgovarajućim direktorijumima čiji naziv je asociran sa jednom od 3 laboratorijske postavke koju je moguće izabrati prilikom rada u laboratoriji (ACIM1, ACIM2 i PMSM) i to u poddirektorijumu LAB01. Detaljnije je moguće uočiti iz Tabele 4.1.

Postavka	Lokacija projektnog direktorijuma	Fajlovi od značaja
ACIM1	D:\..\ACIM1\LAB_IREG\LAB01	firmware.hex (programski kod) plot_IREG_LAB01.m (prikaz grafika) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP)
ACIM2	D:\..\ACIM2\LAB_IREG\LAB01	firmware.hex (programski kod) plot_IREG_LAB01.m (prikaz grafika) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP)
PMSM	D:\..\PMSM\LAB_IREG\LAB01	firmware.hex (programski kod) plot_IREG_LAB01.m (prikaz grafika) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP)

Tabela 4.1 Lista projektnih fajlova

U okviru vežbe nalaze se 2 zadatka (4.4.1. je uvodni računski zadatak dok je 4.4.2. zadatak koji predstavlja praktični rad). Rešenje i rezultate iz ova dva zadatka potrebno je uneti u izveštaj koji se nalazi na kraju vežbe „Podešavanje parametara regulatora struje” (poglavlje 4.5).

Zadatak 4.4.1.

U okviru ove vežbe koristi se GUI (*Graphical User Interface*) prozor putem kog se vrši zadavanje referenci struje kao i prikaz talasnog oblika struje. U cilju olakšavanja korišćenja ovog alata korisno je odrediti rezoluciju mernog sistema struje. Svaka od 3 laboratorijske postavke ima zaseban merni sistem pa se i rezolucije merenja struje međusobno razlikuju.

Šema mernog sistema struje prikazana je na Slici 4.5. U Tabeli 4.2 date su vrednosti elemenata koji su karakteristični za svaku postavku.

	ACIM1	ACIM2	PMSM	Jedinica
$R_{Šinvertor}$	100	100	100	Ω
$R_{Šreg}$	68	47	56	Ω
R_x	10k	10k	10k	Ω
C_x	100	100	100	pF
I_{max}	+/- 9	+/- 22	+/- 42	A
N_{LEM}	250	500	1000	-

Tabela 4.2 Karakteristične veličine u sistemu za merenje struje

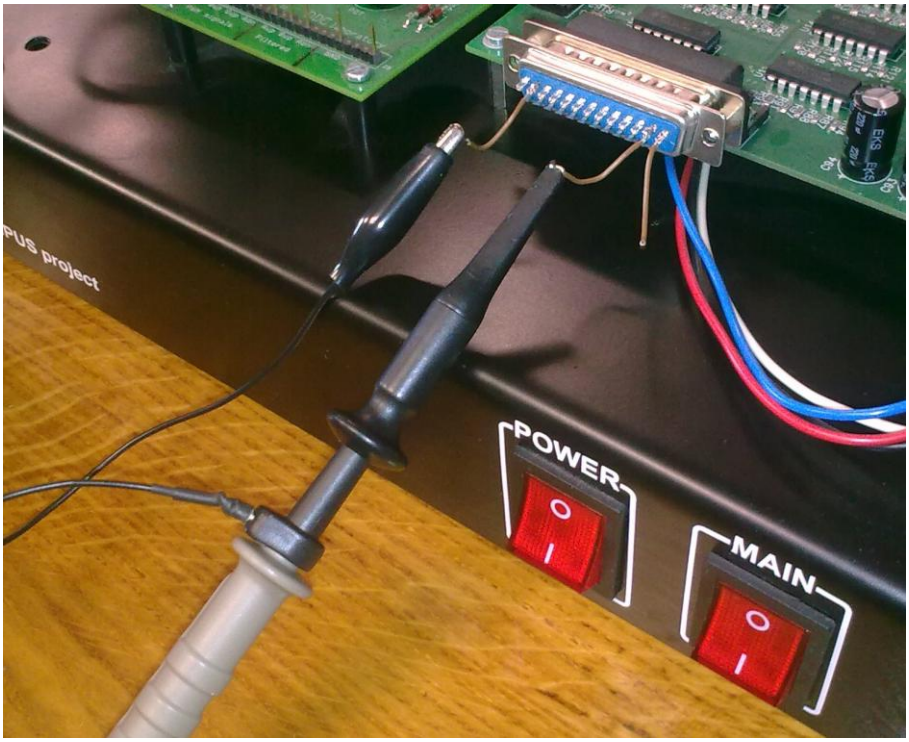
U zavisnosti od postavke na kojoj student radi vežbu, potrebno je izračunati rezoluciju merenja struje. Dobijena rezolucija ima jedinicu A/LSB i predstavlja najmanji kvant struje koji se može detektovati prilikom A/D konverzije. Odgovor uneti u izveštaj koji se nalazi u poglavlju 6.

Zadatak 4.4.2.

Ovim zadatkom počinje praktičan rad u laboratoriji. U ovom zadatku će biti vršeno snimanje signala struje na izlazu iz davača analognim osciloskopom (pre digitalizacije) i putem USB osciloskopa (nakon digitalizacije) na osnovu čega je moguće uočiti rad odabirača kao i uticaj tehnike umanjvanja šuma na signal struje.

Postupak rada vežbe dat je u nekoliko narednih stavki, počevši od 4.2.1. pa do 4.2.8. U koracima je opisan postupak rada vežbe, od načina povezivanja određene opreme, preko programiranja DSP-a pa sve do rada u GUI prozoru. U nekim od stavki nalaze se i pitanja na koja je potrebno odgovoriti (odgovor upisati u izveštaj u poglavlju 4.5.).

4.4.2.1. U saradnji sa dežurnim asistentom priključiti sonde osciloskopa na odgovarajuće pinove izvedene na konektoru (Slika 4.17). Signal koji se meri na ovaj način je izlazni signal iz davača struje (na Slici 5 je to napon na R_s).



Slika 4.17 Merno mesto struje

4.4.2.2. U DSP je potrebno upisati programski kod demo vežbe kao prema uputstvu datom u poglavlju 1 – Demo vežba (u ovom slučaju kod koji se upisuje u memoriju DSP-a je kod koji realizuje regulaciju brzine). Kod se nalazi u fajlu *firmware.hex* koji se nalazi u projektnom direktorijumu (pogledati Tabelu 4.1). Kao podsetnik, u narednih nekoliko stavki, biće ponovljeno skraćeno uputstvo za unos

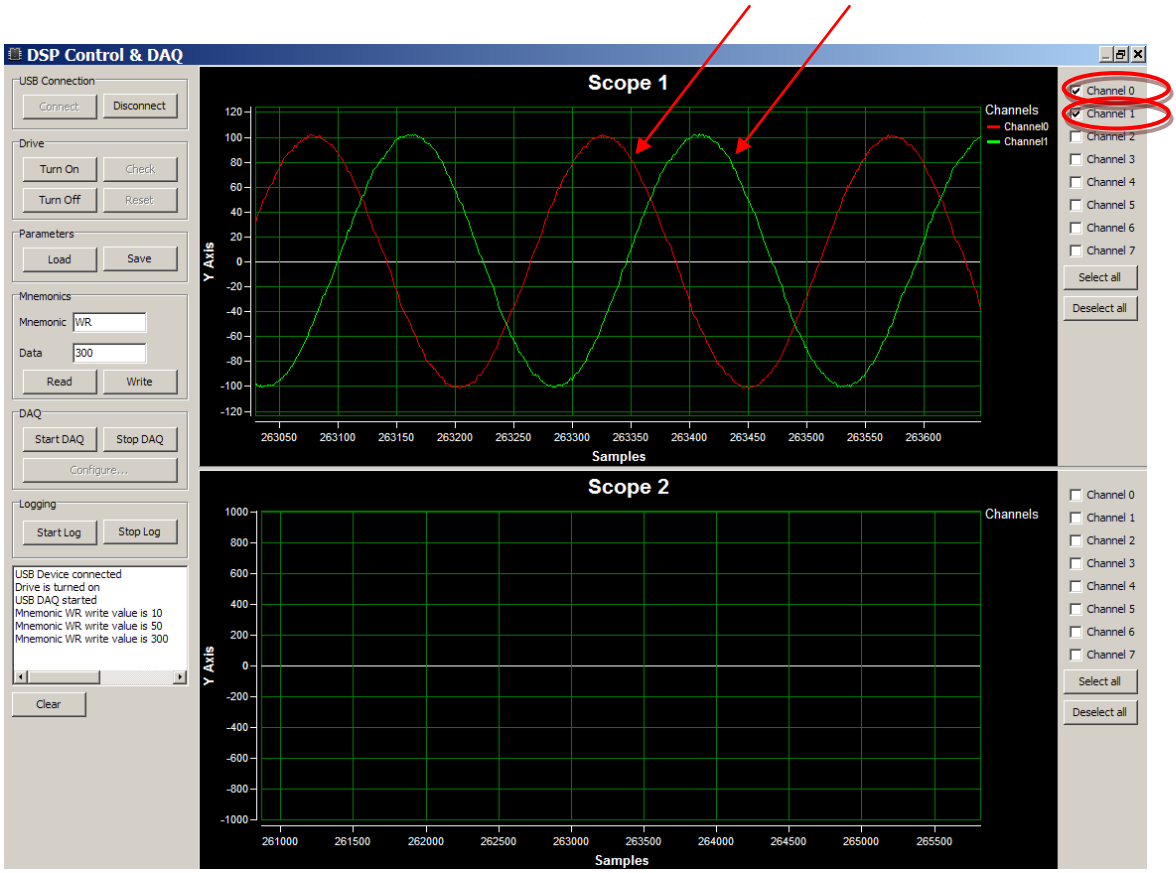
koda u memoriju DSP-a (detaljno uputstvo se nalazi u već pomenutom prilogu). Sledeću proceduru je poželjno savladati jer će se i u drugim vežbama ponavljati.

- Lociranje programskog direktorijuma
- Uključivanje prekidača *MAIN* i *POWER* (prekidači prikazani na Slici 4.17)
- Postavljenja odgovarajućeg džampera na DSP ploči u poziciju koja omogućava unos koda
- Resetovanje DSP-a pritiskom na odgovarajući taster *RESET* koji se nalazi na DSP pločici
- Pokretanje fajla *program.bat* koji se nalazi u projektnom direktorijumu i potom sačekati da se završi unos koda
- Uklanjanje džampera i ponovno resetovanje procesora pritiskom na taster *RESET*

4.4.2.3. U projektnom direktorijumu se nalazi fajl *dspgui.lnk* koji predstavlja prečicu ka GUI prozoru iz koga se zadaju komande pogonu. Potrebno je pokrenuti fajl *dspgui.lnk* a zatim startovati pogon pomoću GUI-a putem sledećih nekoliko koraka (na Slici 4.20 moguće je uočiti izgled prozora i raspored komandi)

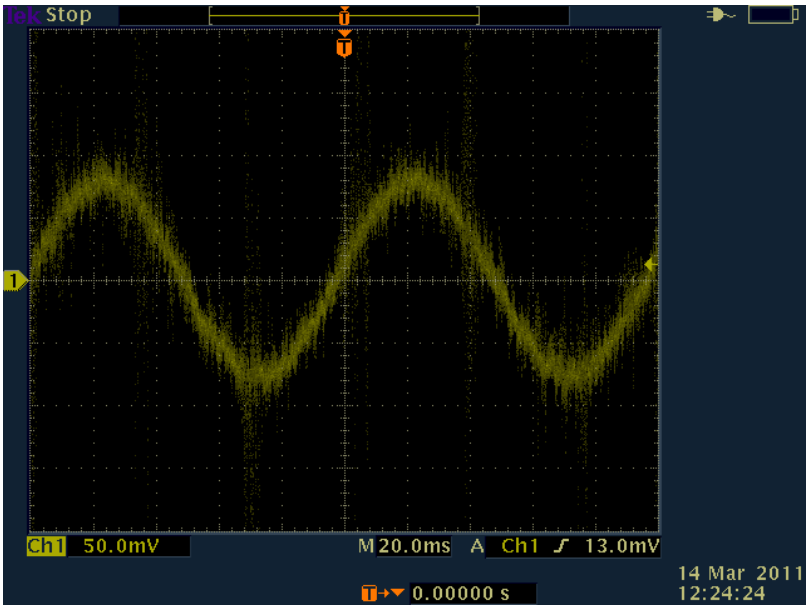
- Pritiskom na polje *Connect*
- Pritiskom na polje *Turn On*
- Pritiskom na dugme *USB Connect*
- Pritiskom na dugme *Start DAQ*
- U polje *Mnemonic* upisati WR
- U polje *Data* upisati vrednost 300 (zadavanje referentne brzine od 300 o/min) i pritisnuti dugme *Write*

Štiklirati kanale 0 i 1 sa desne strana GUI prozora. Na ekranu USB osciloskopa se prikazuju struje i_a i i_b (Slika 4.18).



Slika 4.18 Izgled GUI prozora i prikaz struje

Sonda osciloskopa je priključena na krajeve davača struje i na Slici 4.19 je prikazan signal koji bi trebalo da se vidi na osciloskopu. Uporediti signal prikazan na osciloskopu sa signalom sa USB osciloskopa.



Slika 4.19 Izgled analognog osciloskopa

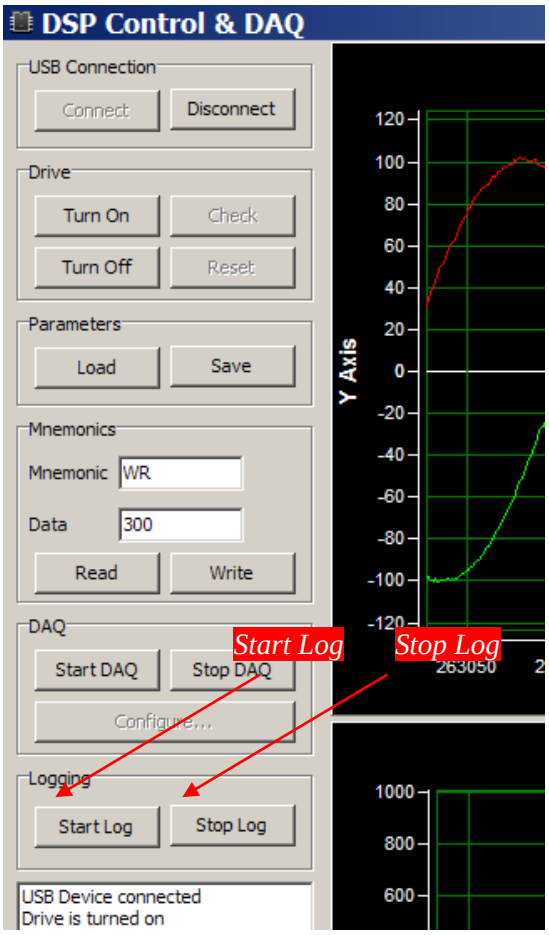
U slučaju da se u situacijama koje na koje asociraju Slike 4.18 i 4.19, prikazuju oblici koji ne odgovaraju oblicima kao na Slikama 4.18 i 4.19 odmah upoznati dežurnog asistenta sa ovom činjenicom.

4.4.2.4. Uporediti nivoe napona na ova dva osciloskopa. Koja ja efektivna vrednost struje faze *a* u ustaljenom stanju pri brzini od 300 o/min (odgovor uneti u izveštaj)? Koristiti odgovor dobijen u stavki 4.4.1.

4.4.2.5. U zavisnosti od postavke na kojoj se radi vežba izračunati faktor G_{LEM} sa Slike 4.16 koji predstavlja faktor pomoću kog se struja pretvara u napon. Za rad ovog zadatka je potrebno i analizirati Sliku 4.5. Odgovor uneti u izveštaj.

4.4.2.6. U zavisnosti od postavke na kojoj se radi vežba izračunati faktor KI sa Slike 4.15. Kojim brojem je predstavljena struja u vrednosti od 1A? Odgovore uneti u izveštaj.

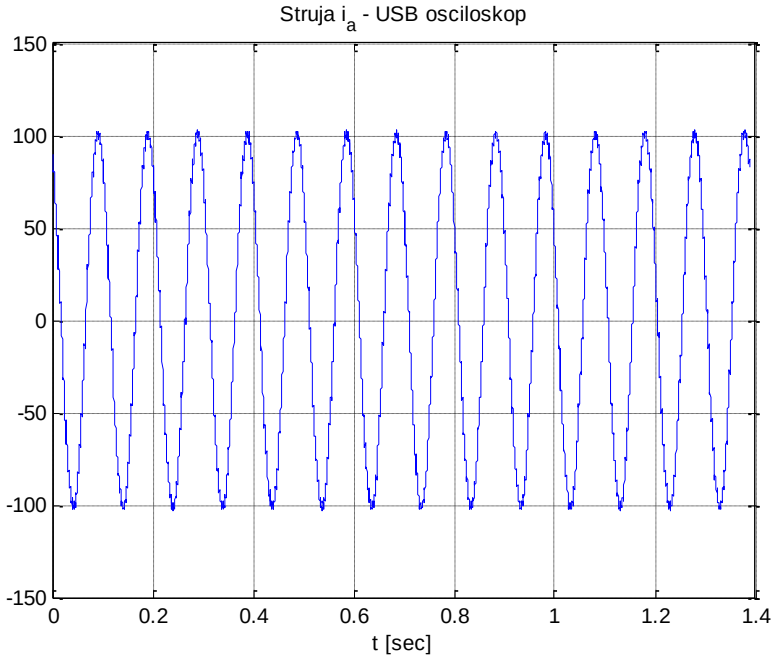
4.4.2.7. Dok se motor obrće zadatom brzinom, u GUI prozoru pritisnuti dugme *Start Log* a odmah zatim i *Stop Log* (pogledati Sliku 4.20 na kojoj ju prikazani pomenuti detalji u GUI-u). Time je signal struja logovan u memoriju PC računara u onolikom trajanju koliko je iznosilo vreme između pritiskanja dugmeta *Start Log* i *Stop Log* (iz tog razloga je potrebno da interval između startovanja i zaustavljanja logovanja bude reda sekunde pošto se manji fajl lakše obrađuje).



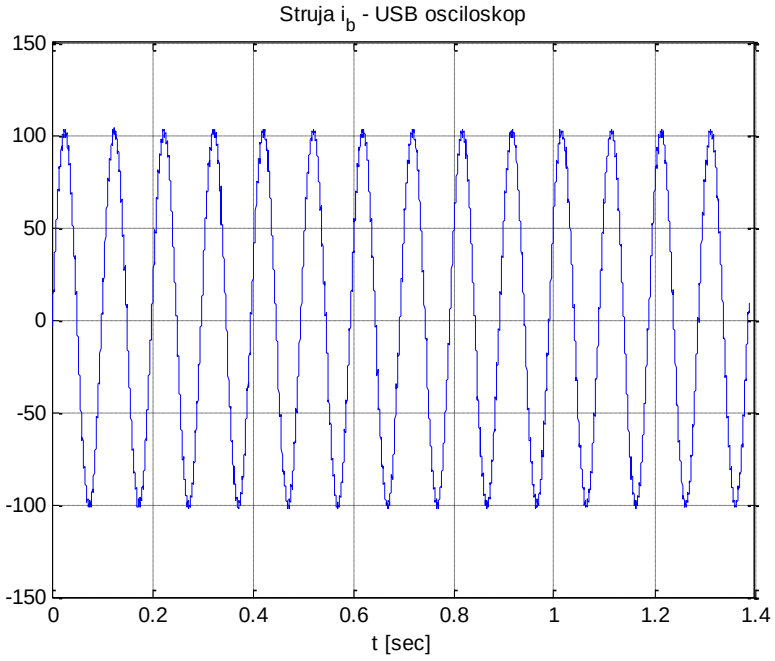
Slika 4.20 Izgled GUI prilikom logovanja podataka

4.4.2.8. Preuzeti od dežurnog asistenta snimljen grafik sa osciloskopa koji predstavlja struju i_a kao i skript u Matlab-u pomoću kog se crtaju grafici. Dobijene grafike struja snimljenim osciloskopom i USB osciloskopom potrebno je odstampati i uneti u izveštaj.

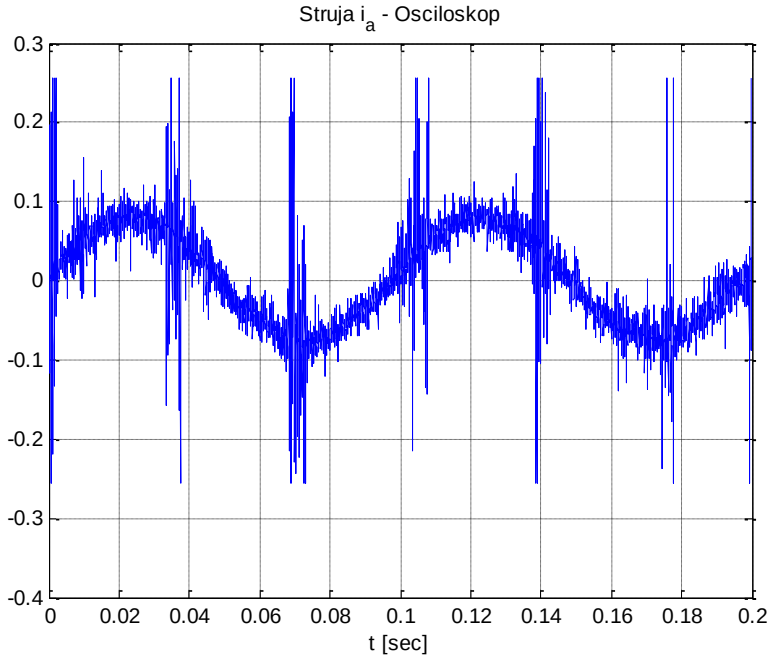
Fajl za prikaz grafika je tako napisan da korisnik nakon pokretanja fajla prvo mora da učita fajl koji je logovan putem GUI-a (sa USB osciloskopa) a potom i fajl koji je dobijen sa osciloskopa. Grafici koji se dobijaju nakon izvršavanja programa bi trebalo da budu nalik slikama 4.21, 4.22 i 4.23.



Slika 4.21



Slika 4.22



Slika 4.23

4.4.2.9. Po završetku rada vežbe zaustaviti pogon pritiskom na polje *Turn Off*. Time će se pogon zaustaviti i invertor neće obezbeđivati napajanje motora.

4.5 Izveštaj

Odgovori:

Student:

Ime: _____

Prezime: _____

Broj indeksa: _____

Laboratorijska postavka _____

4.3. Pitanja za proveru znanja

3.1. (Sinusna look-up tabela) _____

3.2. (PWM) _____

3.3. (PWM) _____

3.4. (Merenje struje) _____

3.5. (Merenje struje) _____

4.4. Laboratorijska vežba „Merenje struje korišćenje LEM davača”

4.1. Zadatak: Rezolucija merenja struje je _____ [A/LSB]

4.2.4. Efektivna vrednost struje u Amperima _____

4.2.5. Faktor G_{LEM} iznosi _____

4.2.6. Faktor skale KI sa Slike 4.15 iznosi _____

4.2.8. Uneti talasne oblike dobijene oscilosopom i USB osciloskopom

Struja i_a (USB osciloskop)

Struja i_b (USB osciloskop)

Struja i_a (osciloskop)

5. Laboratorijska vežba MOT09 5 - Podešavanje parametara regulatora struje

Vežba „Podešavanje parametara regulatora struje“ spada u grupu vežbi iz regulacije struje, kao i vežba MOT09-4. Predmet rada u ovoj vežbi je podešavanje parametara regulatora struje. Kao i u slučaju laboratorijske vežbe u poglavlju 4.4, i za ovu vežbu je potrebno odgovoriti na pitanja koja se nalaze u poglavlju 4.3. i odgovore uneti u izveštaj. Ukoliko je student pitanja rešio pred rad prethodne vežbe u ovom trenutku nema potrebu da se vraća na poglavlje 4.3. Struktura regulatora koji se koristi u laboratorijskoj vežbi „Podešavanje parametara regulatora struje“ nalazi se u poglavlju 4.2.2 ovog uputstva.

Cilj vežbe: sticanje praktičnog iskustva u

- radu sa regulatorom struje na realnom pogonu
- principima podešavanja parametara regulatora struje
- razlikovanju uticaja proporcionalnog i integralnog dejstva regulatora na odziv struje

Detaljniji opis opreme koja se koristi u vežbi se nalazi u Dodatku A i Dodatku B.

Fajlovi potrebni za rad vežbe nalaze se u odgovarajućim direktorijumima čiji naziv je asociran sa laboratorijskom postavkom koju student radi (ACIM1, ACIM2 i PMSM). Detaljnije informacije o fajlovima date su u Tabeli 5.1.

Postavka	Lokacija projektnog direktorijuma	Fajlovi od značaja
ACIM1	D:\..\ACIM1\LAB_IREG\LAB02	firmware.hex (programski kod) plot_IREG_LAB02.m (prikaz grafika) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP)
ACIM2	D:\..\ACIM2\LAB_IREG\LAB02	firmware.hex (programski kod) plot_IREG_LAB02.m (prikaz grafika) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP)
PMSM	D:\..\PMSM\LAB_IREG\LAB02	firmware.hex (programski kod) plot_IREG_LAB02.m (prikaz grafika) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP)

Tabela 5.1 Lista projektnih fajlova

U okviru vežbe nalaze se 2 zadatka. U prvom zadatku je potrebno izvesti određeni račun dok je u drugom zadatku potrebno izvršiti podešavanje parametara regulatora struje. Rešenje i rezultate iz ova dva zadatka treba uneti u izveštaj koji se nalazi na kraju laboratorijske vežbe “podešavanje parametara regulatora struje” (5.3.).

5.1 Određivanje razmere za struju

U vežbi je potrebno izabrati optimalni par vrednosti proporcionalnog i integralnog pojacanja regulatora. Pri tom treba imati predstavu o veličinama koje se predstavljaju na USB osciloskopu. Naime, biće potrebno zadati referentnu vrednost struje od 1 A i to u formi digitalnog ekvivalenta a zatim u zavisnosti od odziva zadavati rezličite parametre regulatora. Pri tome je od velike vežnosti odrediti digitalni ekvivalent struje od 1 A, pa tako u zavisnosti od laboratorijske postavke na kojoj će se raditi podešavanje parametara

regulatora struje, izračunati pomenuti digitalni ekvivalent. Rezutati ovog pitanja se razlikuju u zavisnosti od postavke. Dobijeni broj je potrebno uneti u izveštaj.

Napomena: Ne počinjati rad na stavki 5.2. pre nego što dežurni asistent ne verifikuje rezultat dobijen pod u stavki 5.1.

5.2 Praktičan rad vežbe 5

Ovaj zadatak predstavlja praktičan rad u Laboratorijskoj vežbi 2. U ovom zadatku je potrebno podesiti parametre regulatora struje tako da se ima što povoljniji propusni opseg sa što manjim prebačajem. U narednih nekoliko stavki, počevši od 5.2.1. pa do 5.2.8. dat je detaljan postupak rada vežbe sa potrebnim objašnjenjima.

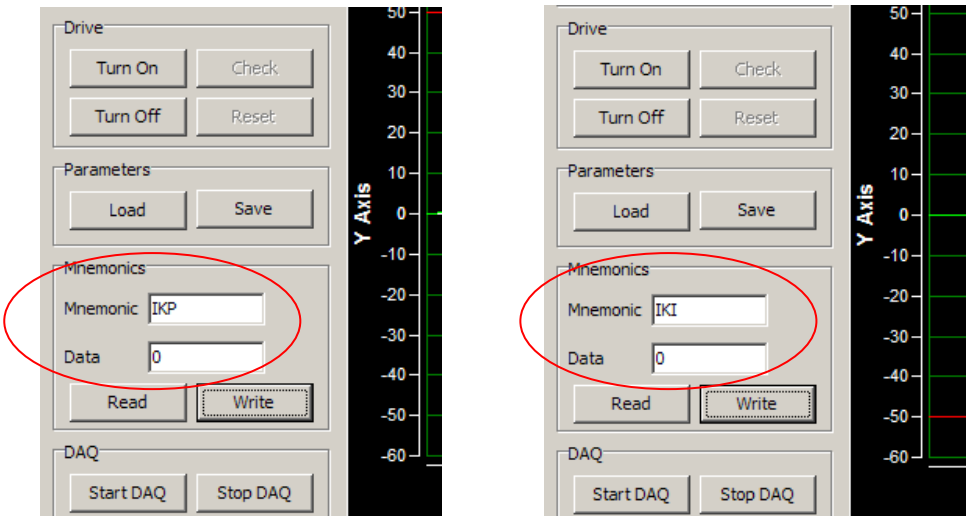
5.2.1. Da bi bilo moguće vršiti regulaciju struje statora motora potrebno je u memoriju DSP-a upisati odgovarajući kod koji realizuje regulaciju na način koji će biti prikazan u nastavku vežbe. U projektnom direktorijumu nalazi se fajl koji je potrebno uneti u memoriju DSP-a. Naziv fajla je *firmware.hex* i predstavlja programski kod za regulaciju struje koji će se koristiti u ovoj vežbi. Postupak unosa pomenutog fajla u DSP je po uzoru na postupak opisan u tački 4.4.2.2. Nakon toga pokrenuti fajl *dspgui.lnk* koji se nalazi u projektnom direktorijumu.

5.2.2. U ovom koraku je prikazan postupak unosa pojačanja regulatora. **Prikazan postupak se primenjuje tek u tački 5.2.4. tako da za sada nije potreban nikakav unos**. Proporcionalno kao i integralno pojačanje je limitirano i može uzimati vrednosti u opsezima prikzanim u Tabeli 5.2.

Dejstvo regulatora struje	*Mnemonic* u GUI-u	Opseg
Proporcionalno	IKP	0 – 10 000
Integralno	IKI	0 – 1 500

Tabela 5.2 Vrednosti u kojima se kreću pojačanja regulatora

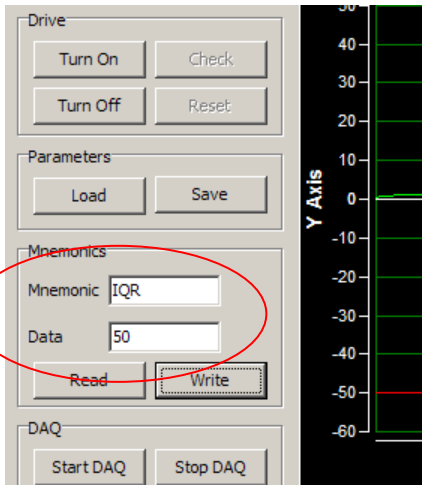
Pojačanja regulatora se unose tako što se u prozor GUI-a u polje *Mnemonic* unese **IKP** za proporcionalno dejstvo ili **IKI** za integralno a nakon toga u polje *Data* se unese vrednost koja mora biti u opsegu koji je prethodno naznačen u Tabeli 4. Nakon unosa vrednosti u polje *Data* potrebno je i potvrditi unos pritiskom na dugme *Write*. Za sada nikakav unos nije potreban.



Slika 5.1 Prikaz unošenja pojačanja regulatora

5.2.3.U ovoj stavki predstavljen je način za unos referentne struje. **Prikazan način će se primenjivati u tački 5.2.4. tako da za sada nije potreban nikakav unos.** Referentna struja koja se zadaje je struja i_q i to tako što se u polje *Mnemonic* upiše **IQR** a u polje *Data* upiše digitalizovana vrednost koja predstavlja struju od 1 A (broj dobijen pod tačkom 5.1.). Izgled GUI prozora dat je na Slici 5.2. Ova referenca struje i_q će se prikazivati na ekranu USB osciloskopa i suština praktičnog rada u ovoj laboratorijskoj vežbi je izbor proporcionalnog i integralnog pojačanja tako da se ima oblik struje koji što je moguće više podseća na zadatu referentnu vrednost (videti Sliku 5.3). Podešavanje regulatora je

dodatno olakšano time što se referenci struje softverski menja znak svakih 5ms pa se ima oblik reference kao crveni trag na Slici 5.3.



Slika 5.2 Prikaz unosa referentne struje24

5.2.4. U ovom koraku je pojašnjen proces podešavanja regulatora struje. **Poželjno je prvo pročitati narednih 5 koraka a nakon toga i praktično početi proceduru podešavanja**.

I. Uskostaviti USB vezu između računara i DSP-a pritiskom na polje *Connect* a zatim uključiti pogon pritiskom na polje *Turn On*

II. Postaviti pojačanja regulatora na 0 (IKP = 0 i IKI = 0, kao na Slici 5.1, koristeći se procedurom iz 5.2.2.)

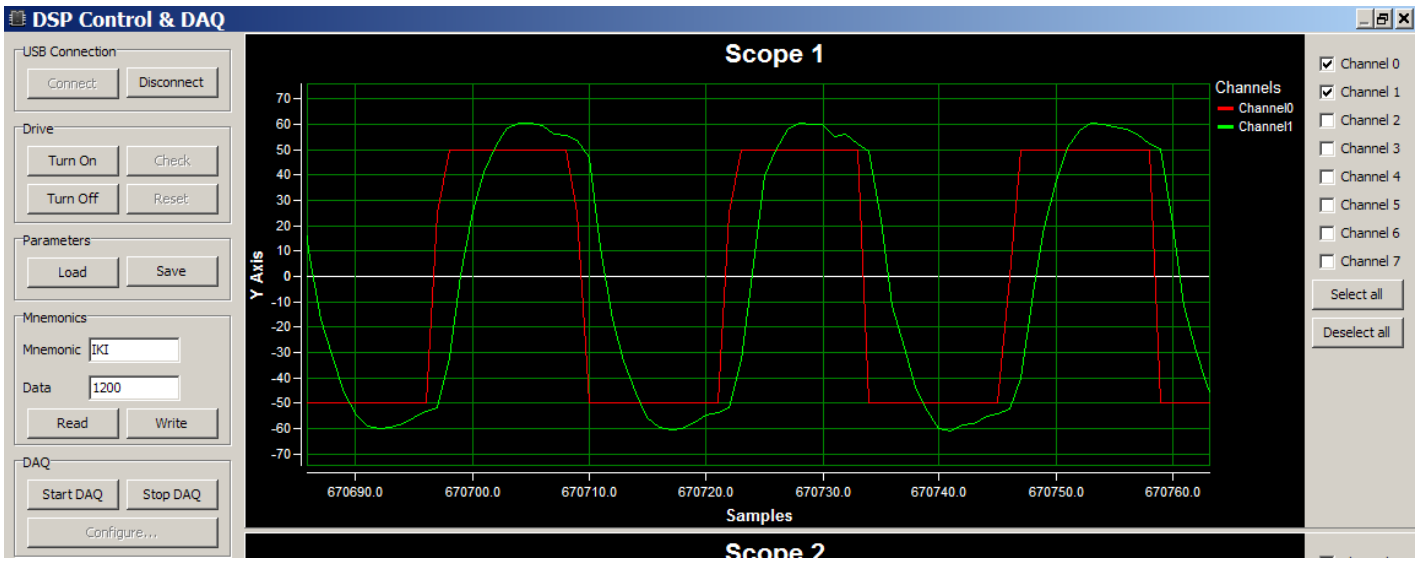
III. Postaviti referentnu struju u *q* osi na vrednost 1 A (kao na Slici 5.3, koristeći se procedurom iz 5.2.3.)

IV. Štiklirati kanal 0 i 1 u GUI prozoru

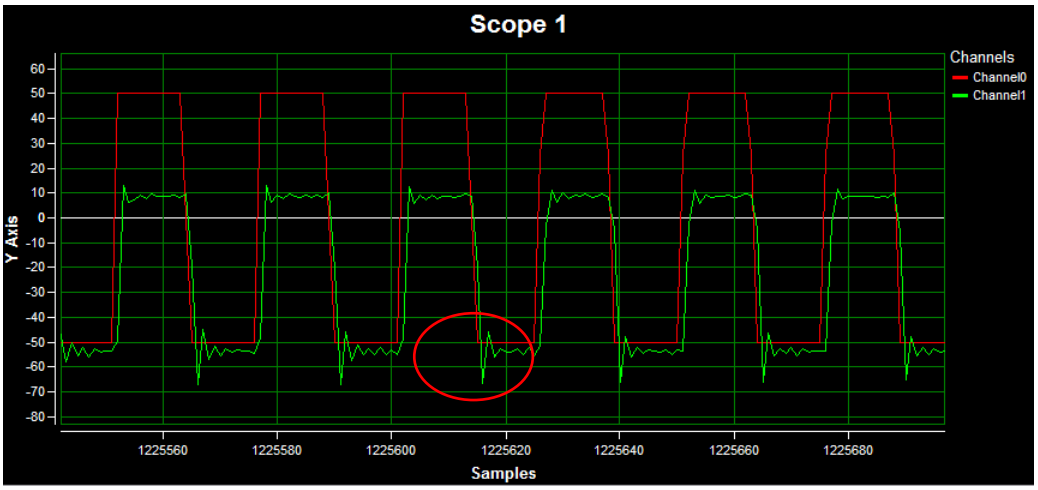
V. Povećavati proporcionalno dejstvo regulatora dok odziv struje ne počne da osciluje oko referentne vrednosti struje

VI. Kada se dogodi oscilovanje opisano pod IV. umanjiti proporcionalno dejstvo za 10-ak % i nakon toga uvećavati integralno dejstvo dok se ne dobije željeni odziv. Primer odziva prikazan je na Slici 5.3.

Prilikom podešavanja proporcionalnog dejstva moguće je startovati od vrednosti 1000 i nastaviti inkrementiranje u koracima od po 200. Sa podizanjem proporcionalnog dejstva treba stati kada se jave oscilacije kao na Slici 5.4 koje su reda 10-20% od referentne vrednosti. Integralno dejstvo, nakon utvrđivanja proporcionalnog dejstva, treba uvećavati počevši od 0 pa u koracima od 50.



Slika 5.3 Referenca i odziv struje



Slika 5.4 Izgled odziva kod aktivnog samo proporcionalnog dejstva

5.2.5. Vratilo motora se prilikom podešavanja regulatora ne okreće. To je omogućeno time što se θ_{dq} kao ulaz u regulator struje, namerno softverski postavlja na 0. Struja injektovana u motor ne stvara obrtno polje već polje koje pulsira oko q ose i samim tim uslovi za uspostavljanje obrtnog momenta nisu ispunjeni. Postavljanje ugla na vrednost 0 je izvedeno prilikom definisanja promenljive koja inače predstavlja ugao u dq transformaciji.

```
volatile unsigned int drive1_theta_s_el = 0; // !!!
```

5.2.6. Nakon podešavanja parametara regulatora, dobijene vrednosti za proporcionalno i integralno pojačanje uneti u izveštaj.

5.2.7. Po uzoru na tačku 4.4.2.7. logovati talasne oblike u što kraćem trajanju. Iz projektnog direktorijuma (LAB02) koristiti priloženi Matlab fajl (plot_IREG_LAB02.m - Tabela 5.1) za prikaz odziva regulatora struje. Grafik uneti u izveštaj.

5.2.8. Proceniti propusni opseg strujne petlje. Dobijenu ocenu propusnog opsega uneti u izveštaj.

5.2.9. Po završetku rada vežbe zaustaviti pogon pritiskom na polje *Turn Off*. Time će se pogon zaustaviti i invertor neće obezbeđivati napajanje motora.

5.3. Izveštaj

Odgovori:

Student:

Ime: _____

Prezime: _____

Broj indeksa: _____

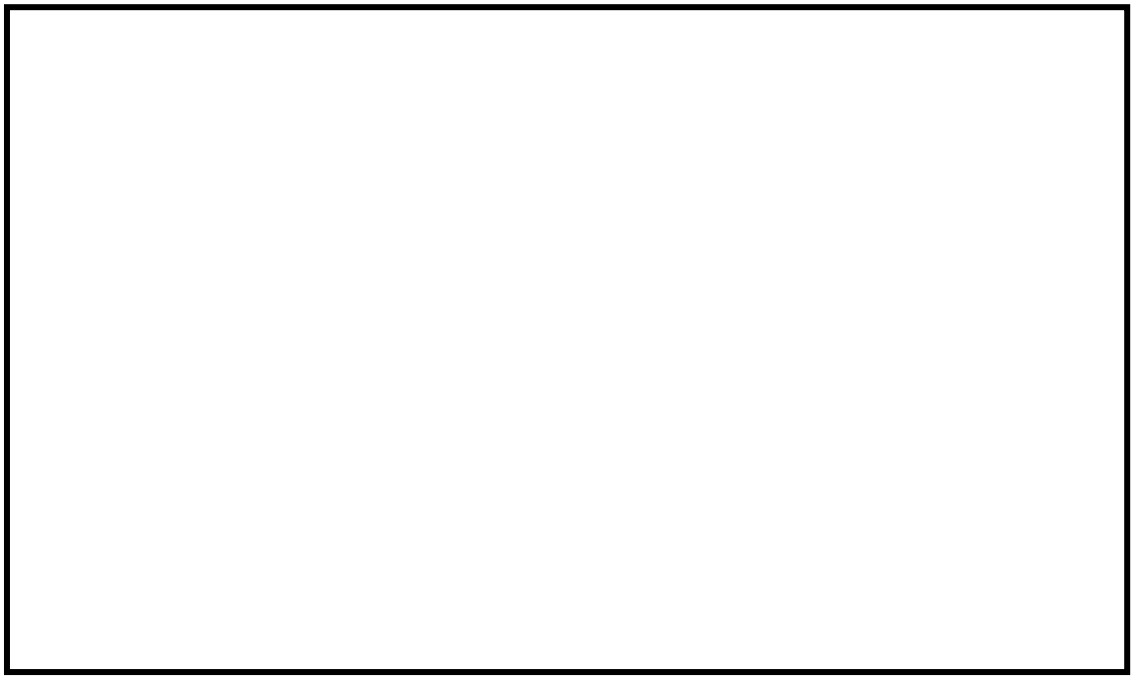
Laboratorijska postavka _____

5. Laboratorijska vežba 2 „Podešavanje parametara regulatora struje“

5.1. Digitalni ekvivalent struje od 1 A je _____

5.2.6. Proporcionalno dejstvo iznosi _____ integralno dejstvo iznosi _____

5.2.7.



Struje i_{qref} i i_q

5.2.8. Procenjeni propusni opseg strujne petlje je _____

6. Laboratorijska vežba MOT09 6 - Podešavanje parametra T_r IFOC strukture

Uputstvo se odnosi na deo laboratorijskih vežbi MOT09 (TEMPUS) u kome je predmet praktičnog rada indirektna vektorska kontrola asinhronog motora. Vežbe se odvijaju u laboratoriji 40a na Elektrotehničkom fakultetu. Pre dolaska na vežbu student treba da pročita ovo uputstvo, kao i dokumente na koje se uputstvo poziva a pre svega prilog koji obrađuje postupak unosa programskog koda u memoriju DSP-a koje se nalazi u poglavlju 1.2 Demo vežba (odeljak 3). Pre rada ove vežbe u laboratoriji, smatra se da je student prošao kroz laboratorijsku vežbu „Digitalna regulacija struje“ gde stečena iskustva mogu dodatno olakšati rad u ovoj laboratorijskoj vežbi.

Indirektna vektorska kontrola se detaljno izučava u okviru predmeta Digitalno upravljanje pretvaračima i pogonima 1. Sve neophodne elemente teorijjskih znanja potrebnih za rad na ovoj vežbi naći čete u delovima Dodatak 1, Dodatak 2, Dodatak 3, Dodatak 4 ovog uputstva. Studentima koji su pratili predavanja nisu potrebne dodatne inormacije i literatura.

Za studente koji nisu pratili kurs Digitalno upravljanje pretvaračima i pogonia 1, detaljniji teorijski osnovi i praktična znanja o digitalnoj regulaciji struje i vektorskom upravljanju mogu se naći

- na sajtu ddc.etf.rs,
- na predavanjima iz OG4DPP, http://ddc.etf.rs/slajdovi10.pdf
- u okviru dokumenta http://ddc.etf.rs/te5mue.pdf (skripta)
- u okviru dokumenta http://emp.etf.rs/epp/mpu_emp/dokt/asmotpos.htm

Detaljniji teorijski osnovi i praktična znanja o digitalnoj regulaciji brzine i položaja naći ćete u udžbeniku

- http://ddc.etf.rs/9781441938541.jpg, *Digital Control of Electrical Drives*

Praktične aspekte vektorskog upravljanja realizovane na pogonu sa analognom regulacijom studenti imaju prilike da vide u laboratorijskoj vežbi VEKTRA. U ovom uputstvu će biti prikazani neki osnovni aspekti indirektne vektorske kontrole na digitalno regulisanom pogonu, najpre kroz kraći teorijski osvrt na model mašine u *dq* sistemu, princip rada algoritma, estimaciju ugla klizanja, merenje pozicije vratila i na kraju će biti predstavljeno nekoliko praktičnih vežbi koje imaju za cilj da studentima predstave rad algoritma indirektne vektorske kontrole (IFOC-a).

Kroz ovo uputstvo student se upoznaje sa

- implementacijom vektorskog upravljanja na digitalno regulisanom pogonu
- estimacijom ugla klizanja
- merenjem pozicije rotora asinhronog motora
- načinom skaliranja varijabli

Detaljniji prikaz opreme koji se koristi u vežbi može se naći u dodacima integralnog uputstva za vežbe i to u odeljcima

Dodatak A,

Dodatak B

6.1. Podešavanje parametra u kalkulatoru klizanja

Indirektno vektorsko upravljanje sadrži kalkulator klizanja u kome figuriše rotorska vremenska konstanta. U okviru vežbe se sprovodi niz merenja na eksperimentalnoj postavci koji za cilj imaju

opredeljenje parametra *slipgain*, koji u kodu predstavlja rotorsku vremensku konstantu i odlučujuće utiče na rad indirektnog vektorskog kontrolera.

Cilj vežbe: sticanje praktičnog iskustva

- u radu sa vektorski kontrolisanim pogonom
- u podešavanju konstante *slipgain*
- u prepoznavanju uticaja konstante *slipgain* na estimaciju ugla klizanja.

Datoteke potrebni za vežbu nalaze se u odgovarajućim direktorijumima čiji naziv je asociran sa laboratorijskom postavkom koju student radi (ACIM1, ACIM2). Za razliku od vežbe „Digitalna regulacija struje“ u ovoj vežbi se ne koristi sinhroni motor sa stalnim magnetima. Studentima su na raspolaganju dve postavke sa asinhronim motorima. Detaljniji prikaz fajlova potrebnih za vežbu dat je u Tabeli 6.1.

Postavka	Lokacija projektnog direktorijuma	Fajlovi od značaja
ACIM1	D:\..\ACIM1\LAB_IFOC\LAB01	firmware.hex (programski kod) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP) plot_IFOC_LAB01.m (prikaz grafika)
ACIM2	D:\..\ACIM2\LAB_IFOC\LAB01	firmware.hex (programski kod) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP) plot_IFOC_LAB01.m (prikaz grafika)

Tabela 6.1

Vežba se sastoji od jednog praktičnog zadatka koji se radi na laboratorijskoj postavci. U zadatku je potrebno podešavati parametar *slipgain* dok se frekvencija statorskih struja ne izjednači sa frekvencijom klizanja.

6.2. Neophodna teorijska znanja

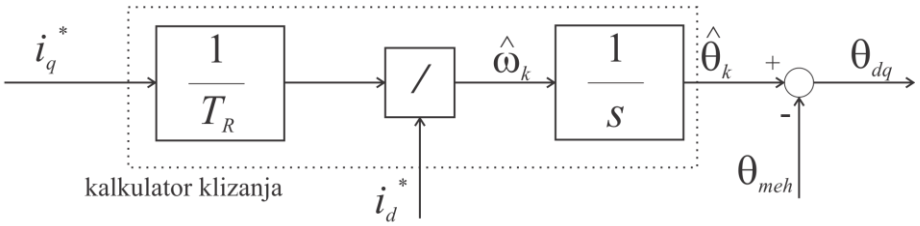
Elemente teorijskih znanja potrebnih za rad na ovoj vežbi mogu se naći u delovima Dodatak 1 (indirektno vektorsko upravljanje), Dodatak 2 (realizacija indirektnog vektorskog upravljanja), Dodatak 3 (merenje pozicije), Dodatak 4 (razmere signala) u okviru ovog uputstva. Detaljniji teorijski osnovi i praktična znanja o digitalnoj regulaciji struje i vektorskom upravljanju mogu se naći

- na sajtu ddc.etf.rs,
- na predavanjima iz OG4DPP, http://ddc.etf.rs/slajdovi10.pdf
- u okviru dokumenta http://ddc.etf.rs/te5mue.pdf (skripta)
- u okviru dokumenta http://emp.etf.rs/epp/mpu_emp/dokt/asmotpos.htm

Predmet rada u ovoj vežbi je podešavanje parametra *slipgain* posmatranjem učestanost statorskih struja na osnovu koje je moguće proceniti da li je parametar dobro usvojen.

6.2.1 Određivanje ugla *dq* sistema

Kalkulator klizanja je dat na slici 6.1. Količnik referentnih struja u d-osi i q-osi se množi sa recipročnom vrednošću rotorske vremenske konstante, $1/T_R = R_R/L_R$. Тако се добија клизање ω_k које одређује брзину предњачења dq система у односу на ротор. Integracijom klizanja se dobija ugao klizanja θ_k. Na slici 6.1, brzina i ugao klizanja imaju znak ^, što znači da se radi o računskim vrednostima koje ne moraju odgovarati stvarnim (primer - ako se unese pogrešna vrednost za $1/T_R$). Kada se na ugao klizanja doda mereni ugao rotora, dobija se položaj dq koordinatnog sistema, koji je jedan od dva preduslova za vektorsko upravljanje.



Slika 6.1 Blok šema sklopa za određivanje ugla θ_{dq}

METOD koji je primenjen tokom vežbanja je sledeći:

- Potrebno je znati nominalnu učestanost klizanja.
- Nasilnom izmenom koda postigne se da je teta_dq = 0
- Sada je ugao dq sistema jednak estimiranom uglu klizanja
- Posmatranjem struja motora meri se učestanost napajanja , to jest, u ovom slučaju, učestanost klizanja.
- Struje id i iq se postave na nominalni iznos.
- Varira se *slipgain* (1/Tr) sve dok se ne ostvari da učestanost bude jednaka nominalnoj učestanosti klizanja.

Ukoliko asinhroni motor radi u oblasti brzina koje su manje od nominalne (u oblasti gde se ne zahteva slabljenje polja) onda se može smatrati da je i_d konstantno pa je ugao klizanja zapravo određen samo komandovanom (referentnom) strujom i_q. Za ugao θ_k je moguće pisati

$$\theta_k = \int_0^t \omega_k \mathrm{dt} \tag{6.1}$$

a pošto se račun ugla klizanja odvija u ekvidistantnim vremenskim intervalima (u laboratorijskim vežbama na svakih 100μs) može se i pisati da je priraštaj ugla klizanja određen strujom i_q. Jednačinom 6.2 prikazan je izraz za određivanje priraštaja ugla klizanja u zavisnosti od struje i_q.

$$\mathrm{d}\theta_k = \omega_k \mathrm{dt} = \frac{1}{T_R}\frac{i_q}{i_d}\mathrm{T} = \frac{\omega_{knom}}{i_{qnom}}\mathrm{T}i_q^* = \mathrm{slipgain}\cdot i_q^* \tag{6.2}$$

Konstanta proporcionalnosti koja predstavlja vezu priraštaja ugla klizanja od referentne *q* struje će se u ovom uputstvu zvati *slipgain*. Prilikom kodiranja u programskom jeziku C za određivanje ugla klizanja koristi se prethodno dobijen izraz.

```
drive1_theta_slip_el.full += drive1_iq_ref * drive1_param_slip_gain; // dodavanje inkrementa
                                                                    //klizanja na prethodni ugao klizanja
```

Navedene linije koda koji računa ugao klizanja nalaze se u fajlovima koji se nalazi u projektnim direktorijumima. U okviru fajlova nalazi se dokument *control.c*, koji sadrži C kod koji realizuje računanje ugla klizanja.

Drive1_theta_slip_el.full je promenljiva koja predstavlja 32-bitni zapis ugla klizanja koji se dobija tako što se u svakoj prekinoj rutini uvećava za definisani priraštaj (jednačina 6.2) koji predstavlja proizvod struju u *q* osi (drive1_iq_ref) i parametra slipgain (drive1_param_slip_gain). Prilikom računanja ugla klizanja potrebno je primeniti 32-bitni zapis iz razloga što je u proračun uključeno množenje 16-bitnih brojeva što za rezultat daje 32-bitni broj.

6.3. Programska implementacija bloka „kalkulator klizanja“

U narednom poglavlju biće prikazan način merenja brzine rotora a nakon tog poglavlja će biti detaljnno prikazane razmere brojeve koje se koriste u određivanju ugla *dq* sistema.

Varijable koje postoje u programu su

drive1_theta_r_el,

koja označava električni ugao rotora, to jest, položaj rotora pomnožen brojem pari polova, i koji (više u Dodatku 4) promenu ugla 0-2π predstavlja promenom 0-0xFFFF

drive1_theta_slip_el,

koja označava električni ugao klizanja. Zbog tačnosti u računu, ova varijabla je 32-bitna, pri čemu se ona može posmatrati kao spoj dva 16-bitna broja, koji su gornji i donji deo 32-bitnog broja. Gornji deo zapravo predstavlja vrednost 32-bitnog broja Kada se posmatra godnji 16-bitni deo 32-bitnog broja, promenu ugla 0-2π predstavlja promena 0-0xFFFF

drive1_param_slip_gain,

koja označava pojačanje slipgain, odnosno, broj koji zavisi od rotorske vremenske konstante. Upravo ovaj broj treba odrediti u okviru vežbe kako bi algoritam ispravno funkcionisao.

drive1_iq_ref,

koja označava referentnu vrednost struje iq (više u Dodatku 4).

Matematički opis kalkulatora klizanja je sledeći, svakih sto mikrosekundi izvrši se proračun nove vrednosti referentne struje iq, izvrši se očitavanje nove pozicije rotora (više u Dodatku 3 kao i u poglavlju 2.7 integralnog uputstva) i sprovodi se sledeći račun,

drive1_theta_slip_el = drive1_theta_slip_el + drive1_iq_ref * drive1_param_slip_gain;

drive1_theta_s_el = drive1_theta_r_el + drive1_theta_slip_el / 65536;

Cilj ove vežbe je odredđivanje drive1_param_slip_gain tako da se osa d uvek poklopi sa rotorskim fluksom. Ova promenljiva se kolokvijalno zove *slipgain*.

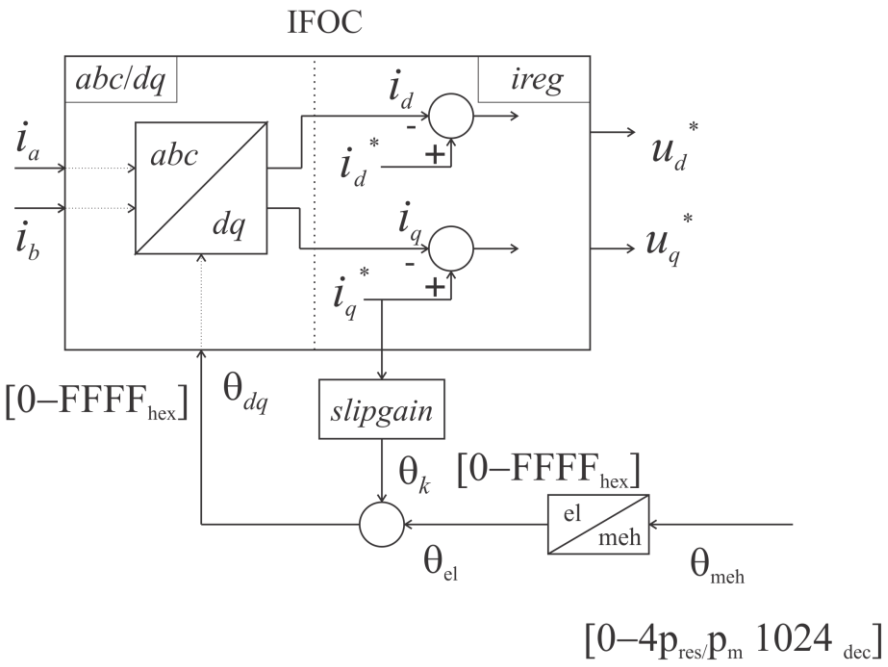
Konstanta *slipgain* se može izračunati na više načina. Ovde se razmatra postavka ACIM2, a motorom koji ima 2 para polova kao i nominalnu brzinu od 1410 o/min pa mu je pri nominalnom režimu rada učestanost klizanja 3 Hz. Ovaj motor imanominalnu vrednost struje koja se u programu predstavlja brojem 123 (više u Dodatku 4). Broj *slipgain* pomnožen sa 123, i dodat na 32-bitni broj **drive1_theta_slip_el** treba da rezultuje učestanošću klizanja od 3 Hz. Dakle, gornji 16-bitni deo ove 32-bitne varijable treba da se uveća za 3 * 65536 svake sekunde. Naime, pozicija (0-2π) se prikazuje sa punim opsegom 16-o bitnog broja (2^{16}-1 = 65535 = 0x0ffff). Dakle, 32-bitni broj se mora inkrementirati za 3 * 65536 * 65536. Svakih 100 mikrosekundi, potrebno je dodati inkrement od 3 * 65536 * 65536 * 0.0001 = 1 288 490 = *slipgain* * i_q*nom*. Zaključuje se da broj *slipgain* treba da bude 1 288 490 / 123 = 10475.

Ovo razmatranje se može potkrepiti sledećom jednačinom. Za prikaz jednog radijana ima 10430. Koristeći se jednačinom 6.2 može se pisati jednačina 6.3

$$slipgain = \frac{\omega_{knom}}{i_{qnom}} \mathrm{T} = \frac{\omega_{knom}}{i_{qnom}} \frac{2^{16}}{2\pi} \mathrm{T} 2^{16} = \frac{3[\mathrm{Hz}]2\pi}{123} \frac{2^{16}}{2\pi} 10^{-4} 2^{16} = 10475_{dec} \qquad (6.3)$$

gde broj $2^{16}/2\pi$ predstavlja rezoluciju prikazivanja jednog radijana, 10^{-4} predstavlja periodu izvršavanja algoritma a 2^{16} predstavlja tačnost zapisa broja (32bit). Tako se za zapis konstante slipgain u C kodu koristi podatak dvostruke tačnosti i u njega se upisuje broj 10475 (više u Dodatku 4).

Razmere brojeva prikazane su na sledećoj slici (Slika 6.2) gde se može uočiti koji brojevi se koriste u algoritmu. Razmere struja i napona koji se imaju u algoritmu mogu se videti u poglavlju 4.2.5.



Slika 6.2 Prikaz razmera pozicije koji se koristi

Blok koji daje ugao *dq* sistema treba da skalira dobijeni mehaniči ugao i da na tako dobijeni ugao nadoda ugao klizanja. Taj blok se u jeziku C vrlo lako kodira. Deo koda koji vrši pomenutu funkciju dat je u nekoliko sledećih linija (ostatak koda koji vrši IFOC regulaciju nalazi se u projektnim diektorijumima o koji su navedeni u Tabeli 6.1).

```
                    // Skaliranje izmerene pozicije vratila motora u elektricni ugao
                    // theta_r_el = 0 - 2 * pi <=> 0 - 0xFFFF
drive1_theta_r_el = drive1_enc << drive1_param_scale_theta_r;
                    // Racunanje ugla klizanja.
                    // theta_slip : 0 - 2 * pi <=> 0 - 0xFFFF
drive1_theta_slip_el.full += drive1_iq_ref * drive1_param_slip_gain;
                    // theta_s (theta_dq): 0 - 2 * pi <=> 0 - 0xFFFF,
                    // theta_s = theta_r_el + theta_slip;
drive1_theta_s_el = drive1_theta_r_el + drive1_theta_slip_el.part.high;
```

Navedene linije koda koji računa ugao θ_{dq} nalaze se u fajlovima koji se nalazi u projektnim direktorijumima. U okviru fajlova nalazi se dokument *control.c*, koji sadrži C kod koji realizuje računanje ugla klizanja.

6.4. Postupak

Da bi se podesio i verifikovao parametar slipgain, koristi se naročito izmenjen program u kome se se ugao rotora namerno postavi na 0 (drive1_enc = 0; gde promenljiva predstavlja vrednost ugla očitanu na osnovu enkoderskih impulsa). Ako se sada struje *d* i *q* postave na nominalne vrednosti dobija se pozicija *dq* sistema koja je jednaka uglu klizanja. Brzina obrtanja *dq* sistema će biti jednaka brzini klizanja što znači da će struje faze motora imati učestanost jednaku učestanosti klizanja. Praćenjem učestanosti statorskih struja moguće je eksperimentalno podesiti konstantu slipgain. Navedeno objašnjenje predstavlja osnov rada **laboratorijske** vežbe. U Dodatku 2 jednačina 6.5 koja predstalvja učestanost klizanja, uz pretpostavku da je ugao rotora jednaku nuli, predstavlja takođe i ugao θ_{dq}.

6.5. Pitanja za proveru znanja pred praktičan deo vežbanja.

Pre početka rada laboratorijske vežbe potrebno je odgovoriti na pitanja za proveru znanja kako bi student došao spreman u laboratoriju.

6.5.1. Koja su dva preduslova za ostvarenje vektorskog upravljanja?

6.5.2 (IFOC) Na koji način se dolazi do potrebnog ugla *dq* sistema u realizovanom vektorskom uptavljanju na laboratorijskim vežbama? Odgovor upisati u poglavlju 6 gde se nalazi izveštaj rada u laboratoriji koji treba popuniti.

6.5.3. (Merenje pozicije) U zavisnosti od postavke na kojoj student radi vežbu, potrebno je da izračuna koliko se ima enkoderskih impulsa faze A za jedan električni period struje statora motora. Odgovor uneti u izveštaj.

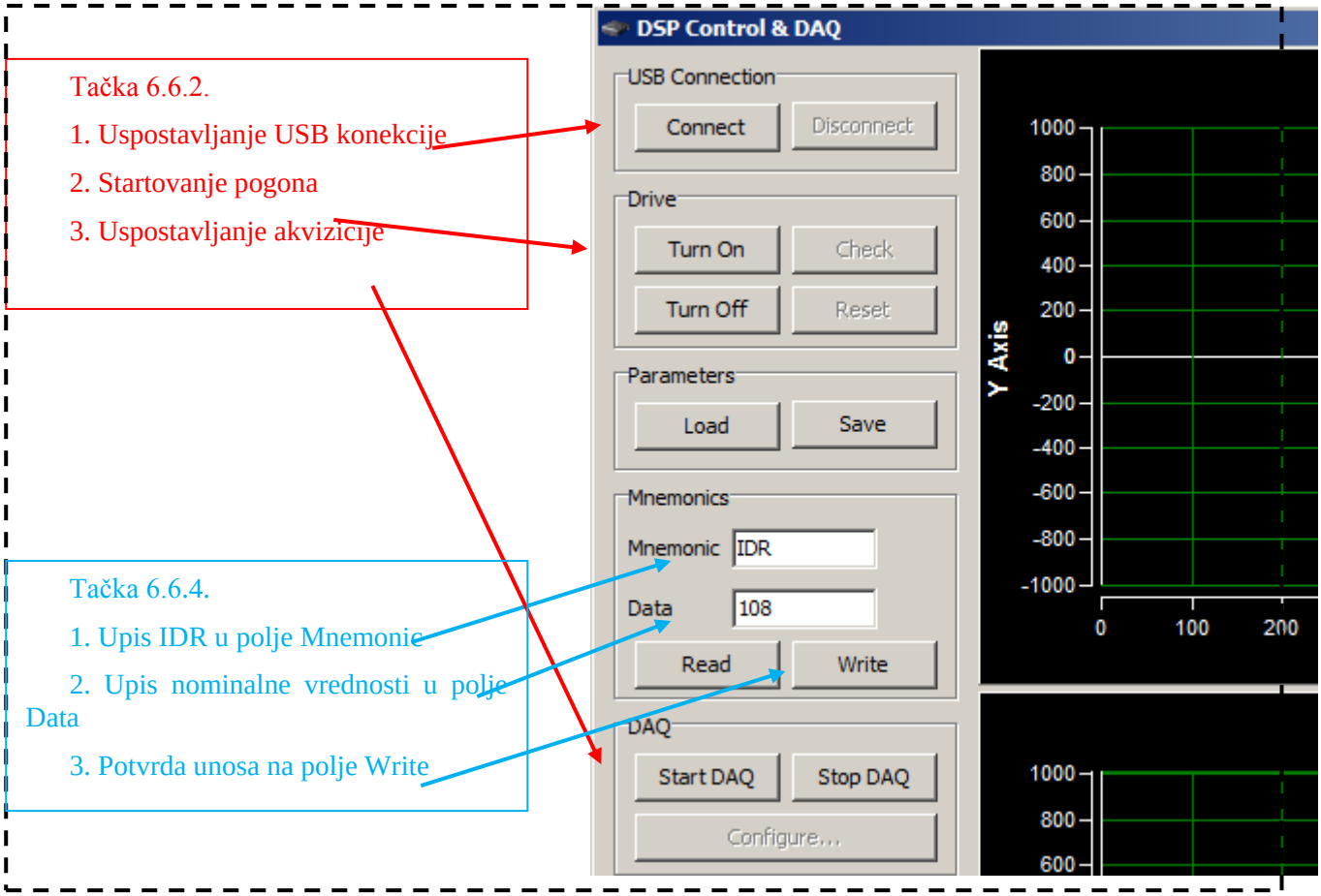
6.5.4. (Merenje pozicije) U zavisnosti od postavke na kojoj student radi vežbu, potrebno je da izračuna kojim brojem treba da se skalira pozicija tako da se koristi pun opseg brojeva (0-FFFF).

6.6. PRAKTIČAN DEO VEŽBE MOT09-6

U narednih nekoliko stavki (6.6.1. do 6.6.9) data je procedura rada u laboratoriji, koja uključuje i vodič u radu sa grafičkim prozorom.

6.6.1. Od ove stavke pa do stavke 6.6.9. se odvija praktični rad laboratorijske vežbe. Da bi bilo moguće raditi vežbu neophodno je upisati odgovarajući programski kod u memoriju DSP-a. Fajl koji treba upisati u DSP se zove *firmware.hex* i nalazi se u projektnom direktorijumu (videti Tabelu 6.2). Proces upisa programskog koda u DSP je korišćen u nekoliko navrata, pod stavkom 4.4.2.2. kao i u poglavlju 1.2 Demo vežba.

6.6.2. Nakon unosa koda, potrebno je pokrenuti fajl *dspgui.lnk* koji predstavlja prečicu kao GUI prozoru. Fajl se nalazi u projektnom direktorijumu. Nakon toga startovati pogon pritiskanjem na polja *Connect*, *Turn On* i *Start DAQ* i to navedenim redom (videti Sliku 6.3).



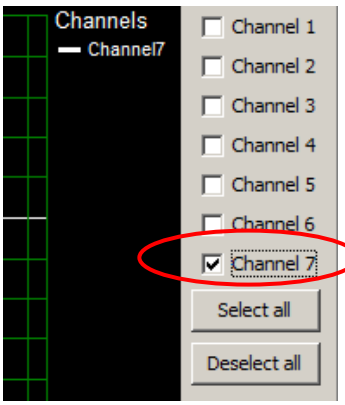
Slika 6.3 Izgled ekrana osciloskopa i procedura korišćenja za stavke 6.6.2 i 6.6.4

6.6.3. Uočiti da se nakon aktiviranja akvizicije signala, apscisa na gornjem i donjem osciloskopu osvežavaju i kontinualno prikazuju različite odbirke. U periodu od 1 sec, prikaže se 2500 odbiraka (semplova).

6.6.4. Referentnu struju i_d je potrebno postaviti na nominalnu vrednost upisom u polje *Mnemonic* „IDR“ i upisom vrednost iz Tabele 1 u polje *Data*. Zatim potvrditi unos pritiskom na *Write* (pogledati Sliku 6.3).

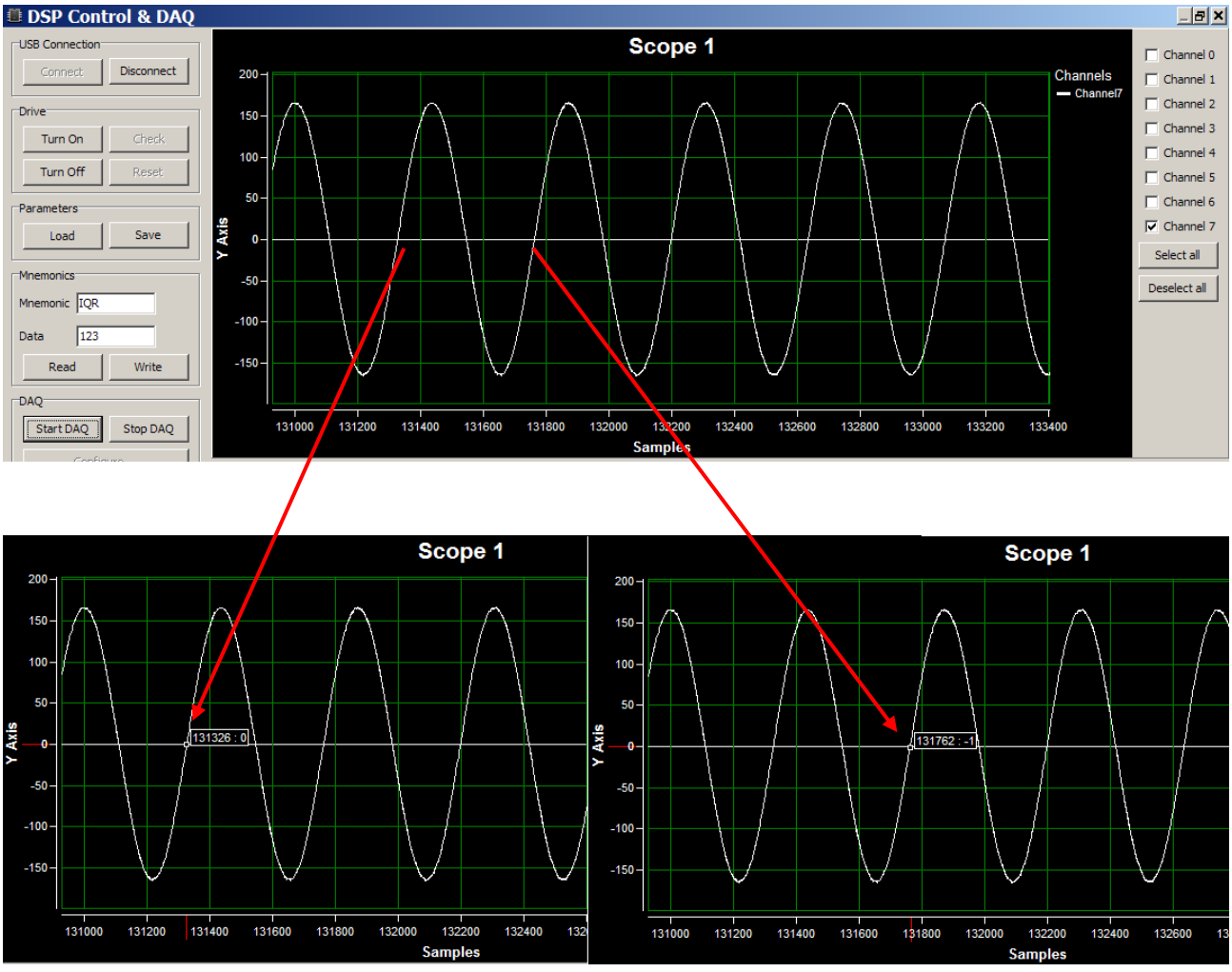
6.6.5. Zatim je potrebno postaviti struju u *q* osi na nominalnu vrednost po istom principu kao i u prethodnoj tački 6.6.4. s tim što se unosi vrednost nominalne i_q struje navedene u Tabeli 6.2.

6.6.6.Potrebno je selektovati kanal 7 (Slika 6.4) u cilju prikazivanja struje faze *a*.



Slika 6.4 Izgled GUI prozora sa selektovanim kanalom

U okviru osciloskopa će se prikazati struja faze *a*.



Slika 4.5 Izgled osciloskopa u slučaju prikaza struje i_a kao i jedan od načina određivanja vremena

6.6.7. Sada je potrebno zaustaviti osvežavanje grafika prtiskom na dugme F1 tastature računara i osciloskop će se zamrznuti. Postavljanjem kursora miša na grafik prikazivaće se koordinate izabrane tačke na grafiku, koje predstavljaju odbirak i vrednost signala. Po uzoru na donji deo slike 6.5, moguće je odrediti vremenski interval trajanja jednog perioda struje faze *a*. U konkretnom slučaju prikazanom na slici 6.5, početni i krajnji semplovi (trenuci) odabranog perioda su 131326 i 131762, što uz objašnjenje dato u tački 6.6.3. daje period od 432 sempla. To odgovara periodu od 0.1728 sekundi što je oko 5.8 Hz. **To je frekvencija dobijena sa početnom vrednošću konstante *slipgain* koja nije odgovarajuća**.

6.6.8. U Tabeli 6.2 uočiti frekvenciju klizanja. Započeti podešavanje konstante *slipgain* sa ciljem da se dobije frekvencija statorske struje koja odgovara frekvenciji klizanja. Procedura promene konstante

slipgain je ista kao i promene, recimo struje i_d što je opisano u tački 6.6.4. tako što se u polje *Mnemonic* unese „SG“ a u polje *Data* željena vrednost. Ovaj proces ponavljati dok se ne dobije odgovarajuča frekvancija statorske struja. Dobijenu vrednost *slipgain* uneti u odgovarajuće polje u izveštaju koji se nalazi u poglavlju 6.7.

6.6.9. Nakon podešavanja parametra potrebno je pokrenuti logovanje po uzoru na tačku 4.2.7. uputstva za vežbu „Digitalna regulacija struje“ u trajanju od reda 1 sec. Koristeći se matlab fajlom iz projektnog direktorijuma uneti talasni obliki struje statora u izveštaj u odgovarajuče polje izveštaja.

6.6.10. Po završetku rada vežbe zaustaviti pogon pritiskom na polje *Turn Off*. Time će se pogon zaustaviti i invertor neće obezbeđivati napajanje motora.

6.7 Izveštaj

Odgovori:

Student:

Ime: _____

Prezime: _____

Broj indeksa: _____

Laboratorijska postavka _____

6.5. Pitanja za proveru znanja

6.5.1. _____

6.5.2. (IFOC) _____

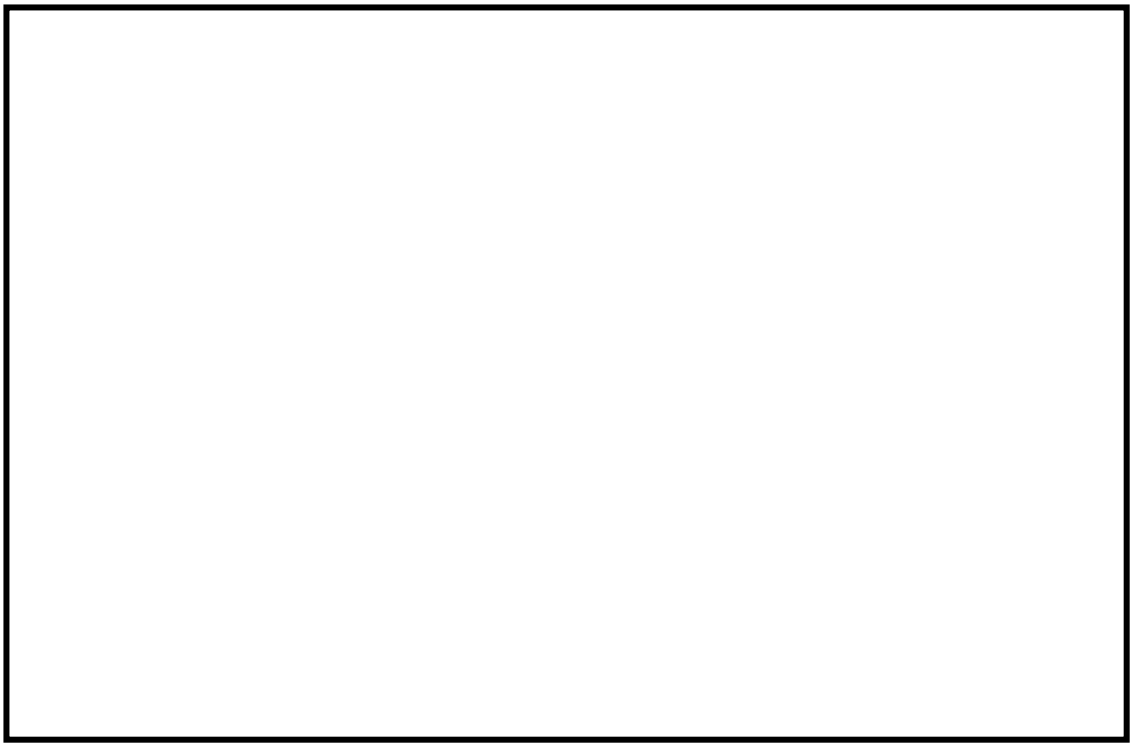
6.5.3. (Merenje pozicije) _____

6.5.4. (Merenje pozicije) _____

6.6. Laboratorijska vežba „Eksperimentalno određivanje konstante *slipgain*"

6.6.8. Konstanta *slipgain* iznosi _____

6.6.9.

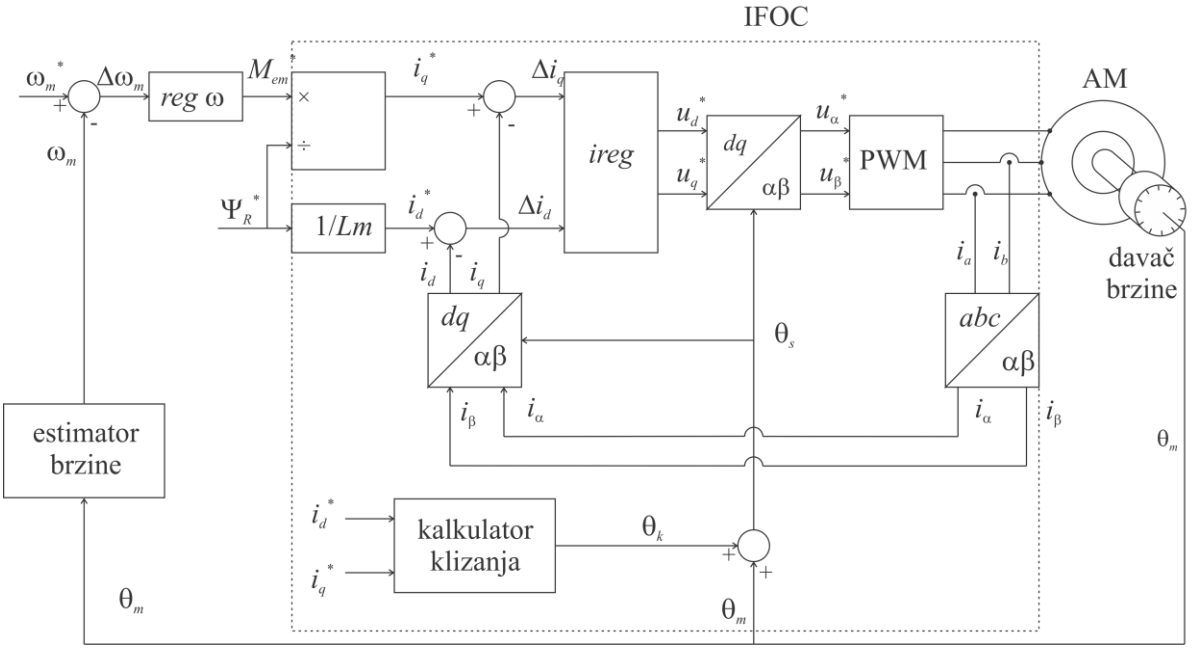


Struja i_a (USB osciloskop)

6.8 Dodatak 1 -Indirektno vektorsko upravljanje

Vektorsko upravljanje predstavlja tehniku kontrole motora putem injektovanja određenih vektora struje u statorske namotaje motora. Za razliku od skalarnog upravljanja, gde je moguće varirati samo amplitudu ili učestanost upravljačkih promenljivih u ovom slučaju se koristi zadavanje amplitude i prostornog ugla upravljačkih promenljivih. Tehnika je primenljiva i kod sinhronih i kod asinhronih motora. Po principu rada, izdvajaju se 2 tehnike upravljanje, direktno i indirektno vektorsko upravljanje i razlikuju se po načinu računanja ugla sinhrono rotirajućeg koordinatnog sistema. U ovoj laboratorijskoj vežbi predmet rada će biti implementacija vektorskog upravljanja na asinhronim motorima i to po principu indirektne vektorske kontorle koja će detaljnije biti predstavljena u ovom uputstvu.

Blok šema vektorski regulisanog pogona se nalazi na Slici 4.6. Na slici je moguće uočiti blokove koji su bili predmet rada laboratorijske vežbe „Digitalna regulacija struje“ i tu se pre svega misli na blokove „ireg“ i „PWM“ koji predstavljaju regulator struje i blok za impulsno širinsku modulaciju, kao i na blokove koji vrše *abc-dq* transformaciju.



Slika 6.6 Blok šema IFOC regulatora koji se koristi u regulaciji brzine

Na Slici 6.6 je prikazana blok šema vektorskog kontrolera koji se koristi u svrhe brzinske regulacije. U osnovi indirektno vektorsko upravljanje vrši regulaciju fluksa i momenta pa se ostavlja mogućnost, kao što je prikazano na Slici 6.6, da se brzinskim regulatorom koji je više hijerarhije, vektorsko upravljanje iskoristi u svrhe regulacije brzine.

U narednom delu uputstva biće predstavljeni neki osnovni detalji koji uz objašnjenja iz vežbe „Digitalna regulacija struje“ stvaraju kompletnu sliku o indirektnom vektorskom upravljanju, što je potrebno za praktični rad.

6.9 Dodatak 2 - Realizacija indirektnog vektorskog upravljanja

U ovom odeljku će biti predstavljena osnovna uloga regulatora koji realizuje indirektnu vektorsku kontrolu (dalje u tekstu će se koristiti skraćenica „IFOC regulator“). U poglavlju 7 dat je u formi priloga detaljniji opis matematičkog modela korišćenog u IFOC regulatoru. Na dalje će se koristiti konačne jednačine koje se dobijaju nakon izvođenja matematičkog modela, predstavljenog u prilogu.

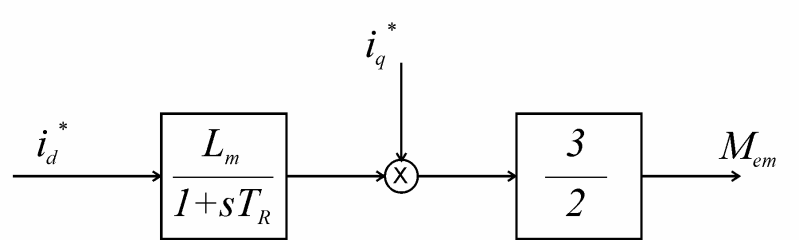
Osnovni zadatak koji treba da reaizuje IFOC regulator je odgovarajuće postavljanje *dq* koordinatnog sistema tako da se osa *d* poklopi sa pozicijom rotorskog fluksa. Tako postavljen *dq* sistem dovodi do toga da se za opisivanje sistema koriste 2 jednostavne jednačine (jednačine 6.4 i 6.5).

$$\frac{\mathrm{d}\Psi_D}{\mathrm{d}t} + \frac{1}{T_R}\Psi_D = \frac{L_m}{T_R} i_d \rightarrow \Psi_D = L_m i_d \tag{6.4}$$

$$\omega_k \Psi_D = \frac{L_m}{T_R} i_q \rightarrow \omega_k = \frac{1}{T_R}\frac{i_q}{i_d} \tag{6.5}$$

Na osnovu navedenih jednačina se zaključuje da je u ustaljenom stanju Ψ_D direktno posledica samo struje statora po *d* osi i da brzina klizanja zavisi od odnosa *q* i *d* komponenata struje statora. Moment motora u ustaljenom stanju se može izraziti jednostavnim izrazom što je prikazano sledećom jednačinom (6.6).

$$M_{em} \approx \frac{3}{2}\left(\Psi_D i_q\right) = \frac{3}{2} L_m i_d i_q \tag{6.6}$$



Slika 6.7 Dinamički model vektorski kontrolisanog motora

Na Slici 6.7 je prikazan blok dijagram kojim je predstavljena dinamika sistema i na osnovu kog se može uočiti da se momenat motora dobija po jednostavnom principu i nalik je blok dijagramu koji se ima kod mašina za jednosmernu struju. Blok dijagram ja dobijen kombinacijom jednačina 6.4 i 6.5.

Treba napomenuti da jednačine i blok dijagram koji su navedeni, formirani su na osnovu pretpostavke da je poznata pozicija rotorskog fluksa kako bi se *d* osa prilikom transformacije koordinate stanja postavila na pravac rotorskog fluksa. Pozicija rotorskog fluksa se kod IFOC algoritma zapravo estimira o čemu je bilo više reči u poglavlju 6.2.

U narednom izlaganju biće predstavljen način za dobijanje ugla koji se koristi u *dq* transformaciji a nakon toga će biti predstavljeni sistem za merenja pozicije i osnovni aspekti praktične implementacije IFOC algoritma.

6.10 Dodatak 3 - Merenje pozicije

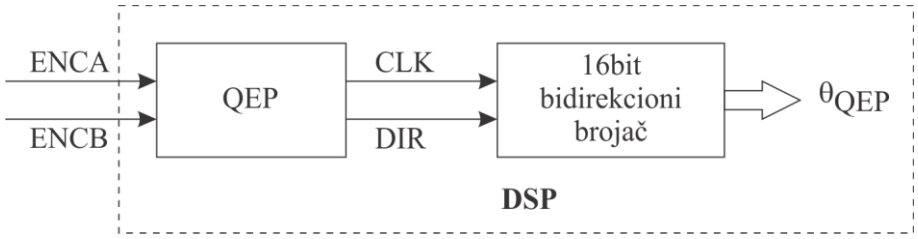
Na laboratorijskim postavkama se za merenje pozicije koriste *resolver*-i. Rezolveri su naprave koje se mogu karakterisati kao električne mašine (sa jednim ili više pari polova). Oni poseduju jedan obrtni transformator koji prenosi visokofrekventni signal (reda 10 kHz) sa statorskog kola na rotorsko kolo. Na statoru se pored sekundarnog namotaja obrtnog transformatora nalaze i detekcioni namotaji na kojima se indukuju naponi koji na neki način zavise od položaja rotora. Detekcioni namotaji su postavljeni normalno jedan u odnosu na drugi i zovu se „sinusni“ i „kosinusni“ namotaj. U njima se indukuju naponi međusobno pomereni za 90 električnih stepeni, frekvencije reda 10 kHz. Na određeni način ovako dobijeni signali se obrađuju sa ciljem da se dobiju enkoderski impulsi to faze A i B (jer se oni veoma jednostvno obrađuju u DSP-u u cilju određivanja pozicije). Funkciju pretvaranja signala rezolvera u enkoderske impulse vrši *Resolver-to-Digital* (*R/D*) konvertor, čip koji se nalazi unutar invertora. Pošto se na vežbama neće zahtevati

detaljno poznavanje senzora pozicije, pomenuti konvertor se neće dalje obrađivati. Više o rezolveru može se čuti na predavanjima iz predmeta Digitalno upravljanje pretvaračima i pogonima 1.

Više o radu sistema za merenje pozicije nalazi se u poglavlju 2.7. ovog uputstva gde se kroz praktičan rad na laboratorijskim postavkama studenti mogu upoznati detaljno sa kompletnim sistemom za merenje pozicije.

Ono što je potrebno znati o sistemu merenja brzine to je da je sistem podešen tako da za jedan električni period rezolvera, R/D konvertor daje 1024 enkoderskih impulsa koji se usmeravaju na dalju obradu u DSP. Odatle sledi da se za jedan mehanički period rezolvera ima p_{res}*1024 impulsa (u jednom mehaničkom obrtaju se ima p_{res} električnih perioda, gde p_{res} predstavlja broj polova rezolvera). U cilju dodatnog pojašnjenja dat je jedan osnovni primer iz prakse: ako se na motoru sa 2 para polova (p_m = 2) primeni rezolver sa 3 para polova (p_{res} = 3) onda se može očekivati da se za 1 električni period struje motora ima p_{res}/p_m*1024 enkoderskih impulsa.

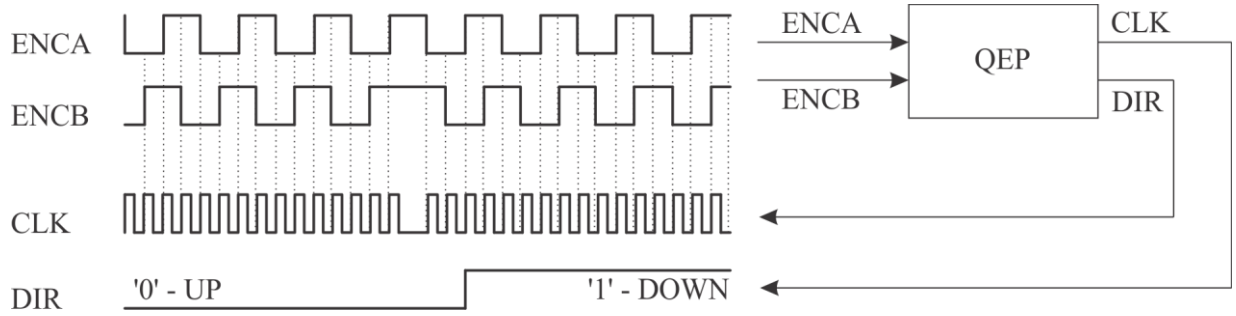
U okviru periferijskih komponenata DSP-a nalazi se i sekcija za brojanje enkoderskih impulsa. Naziv te jednice je *QEP* (Quadrature Encoder Pulse) i u sklopu nje se automatski vrši učetvorostručavanje rezolucije merenja pozicije. Pomenuto učetvorostručavanje proizilazi iz činjenice da u slučaju postojanja 2 faze enkoderskih impulsa koje su pomerene za četvrtinu periode, postoji mogućnost u toku jedne periode impulsa definisati 4 događaja koji definišu inkrement ili dekrement pozicije. Kada *QEP* jedinica zabeleži promenu pozicije šalje impuls za promenu stanja brojača koji je dodeljen *QEP* jedinici i uz to šalje i signal koji definiše znak promene stanja (inkrement/dekrement). Brojač dodeljen *QEP* ima samo jednu ulogu a to je brojanje inkremenata pozicije. Na narednim slikama biće detaljnije predstavljen sistem obrade enkoderskih impulsa kroz prikaz blok šemi kao i kroz analizu signala i razmera.



Slika 6.8 Uproščeni blok dijagram sklopa unutar DSP za merenje pozicije

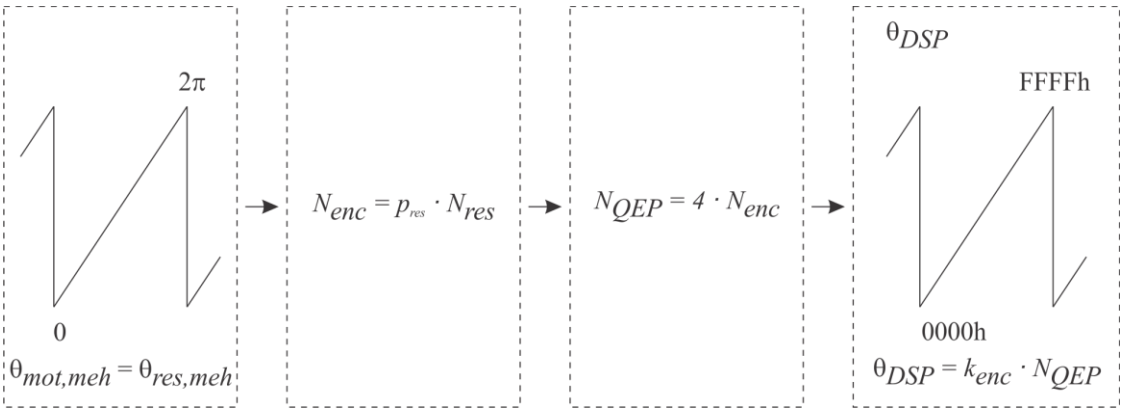
Na Slici 4.8 je blok dijagram sistema za obradu enkoderskih signala dobijenih sa R/D konvertora. Kao krajnji rezultat ima se θ_{QEP}, broj koji sadrži zabeleženi broj inkremenata pozicije. Signal CLK definiše promenu stanja brojača dok signal DIR definiše smer promene.

Na Slici 4.9 je dat prikaz signala koji se imaju pri skokovitoj promeni brzine obrtanja, sa negativne na pozitivnu. Treba uočiti da *QEP* jedinica nakon što detektuje promenu smera obrtanja vratila, menja stanje signala DIR koji definiše smer brojana brojača. Takođe, isprekidanim linijama prikazani su trenuci kada se beleži događaj koji uzrokuje promenu stanja brojača.



Slika 6.9 Prikaz faza enkodera prilikom promene smera brzine

Tok signala koji uključuje i određene faktore skaliranja (koji su opisani u odeljku 2.2 Razmere) dati su na Slici 4.10 kako bi bilo olakšano razumevanje toka signala i skaliranja unutar DSP-a.

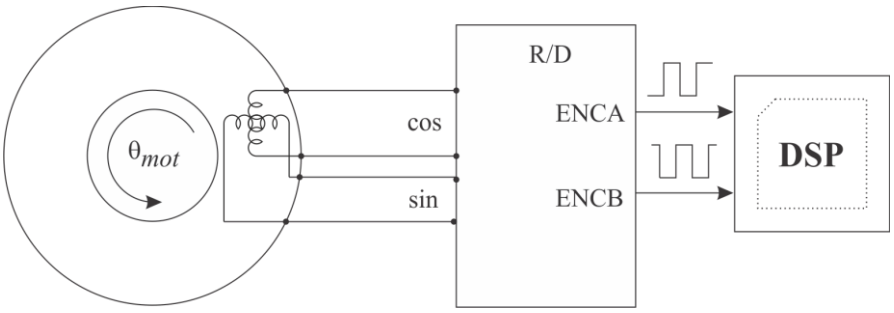


Slika 6.10 Razmere pozicije

Prvi deo Slike 6 prikazuje promenu mehaničke pozicije od 0 do 2 π, drugi i treći deo prikazuju skaliranja koje se imaju (broj enkoderskih impulsa je određen brojem p_{res} dok broj je CLK impulsa koji se ima 4 puta veći od broja enkoderskih impulsa). Na kraju dobijeni ugao θ je potrebno skalirati kako bi se iskoristio pun opseg 16-o bitnog broja).

Više o opisu opreme, o broju pari polova motora i rezolvera koji se nalaze na postavkama moguće je naći u dokumentu koji se bavi prikazom korišćene opreme.

Rezime (merenje pozicije): Na Slici 4.11 data je principijelna šema sistema za merenje pozicije.

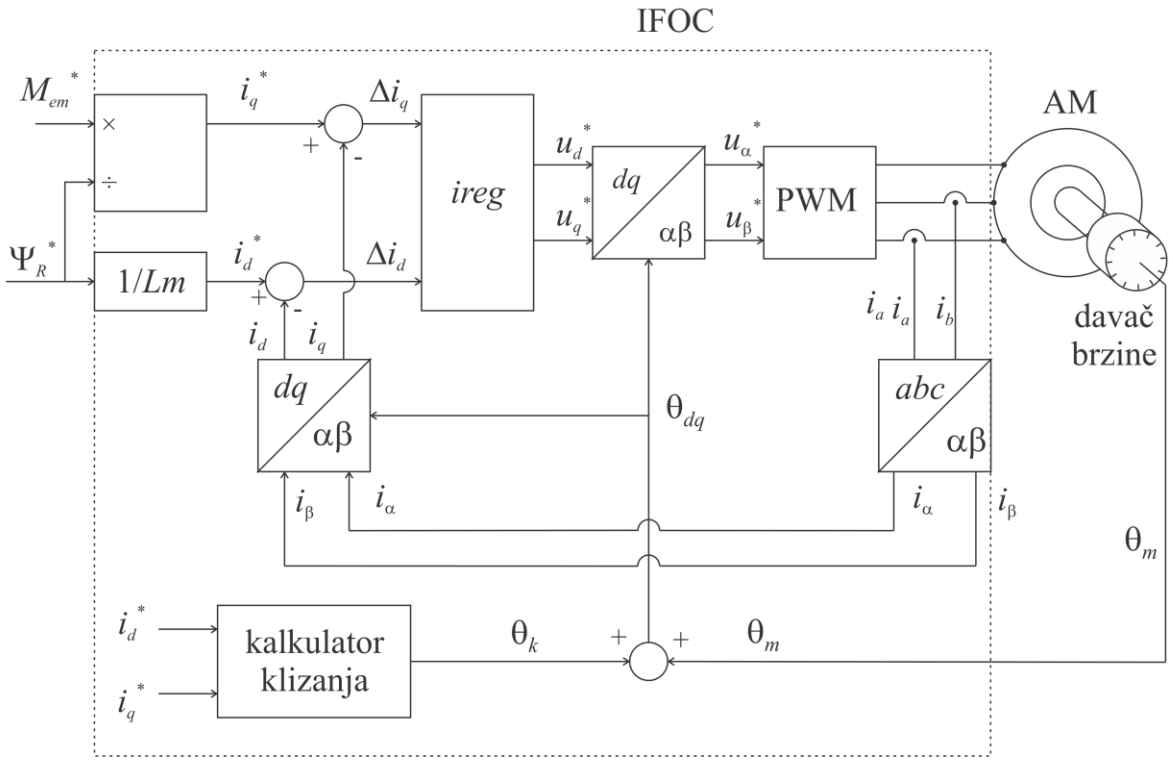


Slika 6.11 Principijena šema merenja pozicije

Obrtni deo davača pozicije se obrće brzinom vratila motora i na osnovu toga generiše određene signale čiji se oblik pretvara u R/D konvertoru. Dobijeni signali imaju formu enkoderskih impulsa i vode se u DSP na dalju obradu. Obradu u DSP-u vrši brojač enkoderskih impulsa. Pristupanjem sadržaju brojača impulsa u ekvidistantnim vremenskim intervalima moguće ja meriti kako poziciju tako i brzinu obrtanja vratila.

6.11 Dodatak 4: Struktura IFOC regulatora i korišćene razmere

Na Slici 6.12 je prikazana struktura IFOC regulatora. Blok „kalkulator klizanja“ ima ulogu u računanju pomenutog ugla i to tako što na izmereni ugao (položaj) rotora nadodaje estimirani ugao klizanja, što za rezultat daje θ_{dq}.



Slika 4.12 Blok šema pogona sa IFOC regulacijom

Blokovi „ireg“, „PWM“ i blokovi koji obavljaju transformaciju koordinata stanja (*abc*-*dq*) obrađivani su u laboratorijskoj vežbi „Digitalna regulacija struje“ i u ovom uputstvu neće biti dalje obrađivani.

Pozicija vratila se meri na osnovu enkoderskih impulsa doijenih sa R/D konvertora. O načinu merenja pozicije više se govori u poglavljima 2.7. i 6.11.

Na predavanjima iz predmeta Digitalno upravljanje pretvaračima i pogonima 1 se detaljnije obrađuje rad sa enkoderskim impulsima. Ono što je najbitniji detalj u ovom procesu je to da jedinica koja broji impulse faze A i B u toku jednog obrtaja, ima mogućnost da prikaže poziciju u 4 puta boljoj rezoluciji nego što je broj impulsa. Kao primer može poslužiti sledeće, ukoliko je broj pari polova rezolvera jednak 2, broj enkoderskih impulsa u toku jednog obrtaja koji se ima je 2*1024 što je 2048. Međutim jedinica u okviru DSP-a koja broji te impulse uspeva da poziciju prikaže u 4*2048 koraka koristeći uzlazne i silazne ivice impulsa.

Tako dobijena pozicija se koristi za računanje brzine i za transformaciju koordinata stanja. Međutim za transformaciju koordinata stanja se ne koristi mehanički ugao već električni. U okviru algoritma pun električni ugao se prikazuje punim opsegom 16-bitnog broja (ugao 0 - 2π mehaničkih stepeni se prikazuje brojevima 0 – FFFF). Iz razloga merenja mehaničkog ugla a postojanjem potrebe za električnim uglom potrebno je izvršiti određeno skaliranje kako bi se iz mehaničkog dobio električni ugao. U konkretnom slučaju, na primer za asinhroni motor iz postavke ACIM2, motor ima 2 para polova dok rezolver ima 1 par polova. Za jedan električni period R/D konvertor daje 512 impulsa, što nakon učetvorostručavanja daje broj 2048 kojim se prikazuje električni ugao. Ovako dobijen broj treba da se prikaže u punom opsegu brojeva (FFFF) pa se skalira brojem 32 (ili šiftuje u levo 5 puta). Slična logika se primenjuje i za postavku ACIM1 samo što se u tom slučaju skalira brojem 16 (šiftuje ulevo 4 puta) jer je rezolver u postavci ACIM1 takav da ima 2 para polova.

U narednoj tabeli dati su osnovni podaci o motorima, njihov zapis u memoriji DSP-a kao i promenljive u programskom kodu koje su asocirane sa određenim veličinama.

	Mnemonic (ukoliko se koristi)	Ime promenljive u kodu (ukoliko se koristi)	ACIM1		ACIM2	
Inom			1,45 A	290_{dec}	1,45 A	208_{dec}
Idnom	IDR	drive1_id_ref	0.97 A	194_{dec}	1.34 A	108_{dec}

Iqnom	IQR	drive1_iq_ref	1.86 A	361_{dec}	1.54 A	123_{dec}
slipgain	SG	drive1_param_slip_gain			1.22 mrad/A	10475_{dec}
Pm			2		2	
Pres			2		1	
f_S			120 Hz*		50 Hz	
n_{nom}			3000 o/min*		1410 o/min	
f_{knom}			3 Hz *		3 Hz	

Tabela 6.2 Lista korišćenih promenljivih i njihove razmere

7. Laboratorijska vežba MOT09 7 - Ispitivanje rada IFOC

Uputstvo se odnosi na deo laboratorijskih vežbi MOT09 (TEMPUS) u kome je predmet praktičnog rada indirektna vektorska kontrola asinhronog motora. Vežbe se odvijaju u laboratoriji 40a na Elektrotehničkom fakultetu. Pre dolaska na vežbu student treba da pročita ovo uputstvo, kao i dokumente na koje se uputstvo poziva a pre svega prilog koji obrađuje postupak unosa programskog koda u memoriju DSP-a koje se nalazi u poglavlju 1.2 Demo vežba (odeljak 3). Pre rada ove vežbe u laboratoriji, smatra se da je student prošao kroz laboratorijsku vežbu „Digitalna regulacija struje“ gde stečena iskustva mogu dodatno olakšati rad u ovoj laboratorijskoj vežbi.

Indirektna vektorska kontrola se detaljno izučava u okviru predmeta Digitalno upravljanje pretvaračima i pogonima 1. Sve neophodne elemente teorijjskih znanja potrebnih za rad na ovoj vežbi naći čete u delovima Dodatak 1, Dodatak 2, Dodatak 3, Dodatak 4 ovog uputstva. Studentima koji su pratili predavanja nisu potrebne dodatne inormacije i literatura.

Za studente koji nisu pratili kurs Digitalno upravljanje pretvaračima i pogonia 1, detaljniji teorijski osnovi i praktična znanja o digitalnoj regulaciji struje i vektorskom upravljanju mogu se naći

- na sajtu ddc.etf.rs,
- na predavanjima iz OG4DPP, http://ddc.etf.rs/slajdovi10.pdf
- u okviru dokumenta http://ddc.etf.rs/te5mue.pdf (skripta)
- u okviru dokumenta http://emp.etf.rs/epp/mpu_emp/dokt/asmotpos.htm

Detaljniji teorijski osnovi i praktična znanja o digitalnoj regulaciji brzine i položaja naći ćete u udžbeniku

- http://ddc.etf.rs/9781441938541.jpg, *Digital Control of Electrical Drives*

Praktične aspekte vektorskog upravljanja realizovane na pogonu sa analognom regulacijom studenti imaju prilike da vide u laboratorijskoj vežbi VEKTRA. U ovom uputstvu će biti prikazani neki osnovni aspekti indirektne vektorske kontrole na digitalno regulisanom pogonu, najpre kroz kraći teorijski osvrt na model mašine u *dq* sistemu, princip rada algoritma, estimaciju ugla klizanja, merenje pozicije vratila i na kraju će biti predstavljeno nekoliko praktičnih vežbi koje imaju za cilj da studentima predstave rad algoritma indirektne vektorske kontrole (IFOC-a).

Kroz ovo uputstvo student se upoznaje sa

- implementacijom vektorskog upravljanja na digitalno regulisanom pogonu
- estimacijom ugla klizanja
- merenjem pozicije rotora asinhronog motora
- načinom skaliranja varijabli

Detaljniji prikaz opreme koji se koristi u vežbi može se naći u dodacima integralnog uputstva za vežbe i to u odeljcima

Dodatak A,

Dodatak B

7.1. Ispitivanje rada IFOC u režimu konstantnog momenta

Indirektno vektorsko upravljanje je u osnovi upravljanje fluksom i momentom motora. U okviru vežbe se ispituje rad algoritma zadavanjem impulsnog momenta i praćenjem brzine obrtanja vratila motora verifikuje

ispravnost algoritma. Motori na kojima se vrše pomenuti eksperimenti nisu opterećeni pa se očekuje linearna promena brzine obrtanja.

Struktura regulatora koji realizuje indirektno vektorsko upravljanje predstavljena je u Dodatku 1, 2, 3 i 4 ovog uputstva.

Cilj vežbe:

- sticanje iskustva u radu sa vektorski kontrolisanim asinhronim motorom
- verifikacija rada algoritma koji implementira upravljanje na osnovu odziva brzine

Fajlovi potrebni za vežbu nalaze se u odgovarajućim direktorijumima čiji naziv je asociran sa laboratorijskom postavkom koju student radi (ACIM1, ACIM2). Za razliku od vežbe „Digitalna regulacija struje“ u ovoj vežbi se ne koristi sinhroni motor sa stalnim magnetima. Studentima su na raspolaganju dve postavke sa asinhronim motorima. Detaljniji prikaz fajlova potrebnih za vežbu dat je u Tabeli 7.1.

Postavka	Lokacija projektnog direktorijuma	Fajlovi od značaja
ACIM1	D:\..\ACIM1\LAB_IFOC\LAB02	firmware.hex (programski kod) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP) plot_IFOC_LAB02.m (prikaz grafika)
ACIM2	D:\..\ACIM2\LAB_IFOC\LAB02	firmware.hex (programski kod) dspgui.lnk (prečica ka GUI prozoru) program.bat (fajl za unos koda u DSP) plot_IFOC_LAB02.m (prikaz grafika)

Tabela 7.1 Lista projektnih fajlova

Vežba se sastoji od jednog praktičnog zadatka u kome se vrši impulsna promena referentne struje *q*.

7.2. Neophodna *teorijska znanja*

Elemente teorijskih znanja potrebnih za rad na ovoj vežbi mogu se naći u delovima Dodatak 1 (indirektno vektorsko upravljanje), Dodatak 2 (realizacija indirektnog vektorskog upravljanja), Dodatak 3 (merenje pozicije), Dodatak 4 (razmere signala) u okviru ovog uputstva. Detaljniji teorijski osnovi i praktična znanja o digitalnoj regulaciji struje i vektorskom upravljanju mogu se naći

- na sajtu ddc.etf.rs,
- na predavanjima iz OG4DPP, http://ddc.etf.rs/slajdovi10.pdf
- u okviru dokumenta http://ddc.etf.rs/te5mue.pdf (skripta)
- u okviru dokumenta http://emp.etf.rs/epp/mpu_emp/dokt/asmotpos.htm

7.3. Postupak

Za ovaj zadatak pripremljen je programski kod koji se nalazi u projektnom direktorijumu i zove se *firmware.hex*. Program je takav da uz održavanje struje i_d na referentnoj vrednosti, svakih 0.2 sekunde menja znak referentne struja u *q* osi koju zadaje student putem GUI prozora. Na taj način se zadaje momenat koji dovodi do sukcesivne promene smera ubrzavanja vratila motora.

Deo koda koji vrši promenu referentne *q* struje je u osnovi identičan kao i u vežbi „Digitalna regulacija struje“ samo što se trenuci promene reference događaju na svakih 0.2 sekunde.

```
if ( ++drive1_ireg_mscounter == 2000 )
{       drive1_iq_ref = -drive1_iq_ref;
        drive1_id_ref = 0;
        drive1_ireg_mscounter = 0;      }
```

Promenljiva drive1_ireg_mscounter predstavlja pomocnu promenljivu koja se inkrementira svakog prekida i kada postane jednaka 2000 (200 msec) referentna struja i_q (drive1_iq_ref) promeni znak. Navedeni deo koda nalazi se u fajlu *control.c* koji se nalaze u projektnom direktorijumu. Ukoliko student želi uvid u pomenuti fajl treba da obavesti dežurnog asistenta.

U Dodatku 2 navedene su osnovne jednačine koje opisuju električni podsistem vektorski kontrolisanog motora. Jednačinom 6.6 i Slikom 6.7 je definisan odziv motora u zavisnosti od struje i_d i i_q. Upravo ovo predstavlja osnovu za laboratorijsku vežbu.

7.4. Praktičan rad

Tok vežbe je dat u narednih nekoliko stavki (7.3.1. do 7.3.11).

7.3.1. Ovom stavkom počinja praktični rad. Za početak je potrebno, po uzoru na tačku 4.2.2. iz uputstva koje obrađuje laboratorijsku vežbu „Digitalna regulacija struje“, uneti programski kod u memoriju DSP-a. Programski kod je unapred pripremljen i nalazi se u fajlu *firmware.hex*.

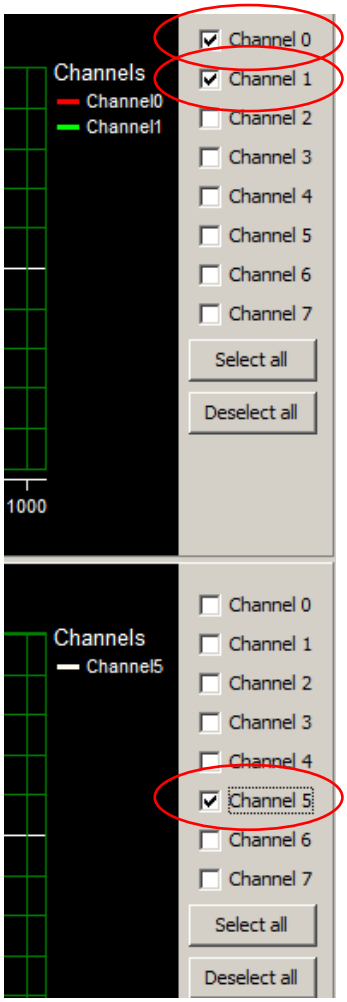
7.3.2. Pokrenuti fajl *dspgui.lnk* koji predstavlja prečicu ka GUI prozoru. Fajl *dspgui.lnk* se nalazi u projektnom direktorijumu svake vežbe.

7.3.3. Aktivirati pogon klikom na polja *Connect*, *Turn On* i *Start DAQ* (Slika 6.3).

7.3.4. Referentnu struju u *d* osi postaviti na 1/3 nominalne vrednosti. Nominalna vrednost struje je zapisana u Tabeli 1. Procedura unosa struje je prikazana u tački 4.1.4. ovog uputstva.

7.3.5. Konstantu *slipgain* postaviti na 3 puta veću vrednost od one koja je dobijena pod tačkom 6.6.8. na način opisan pod tačkom 6.6.4.

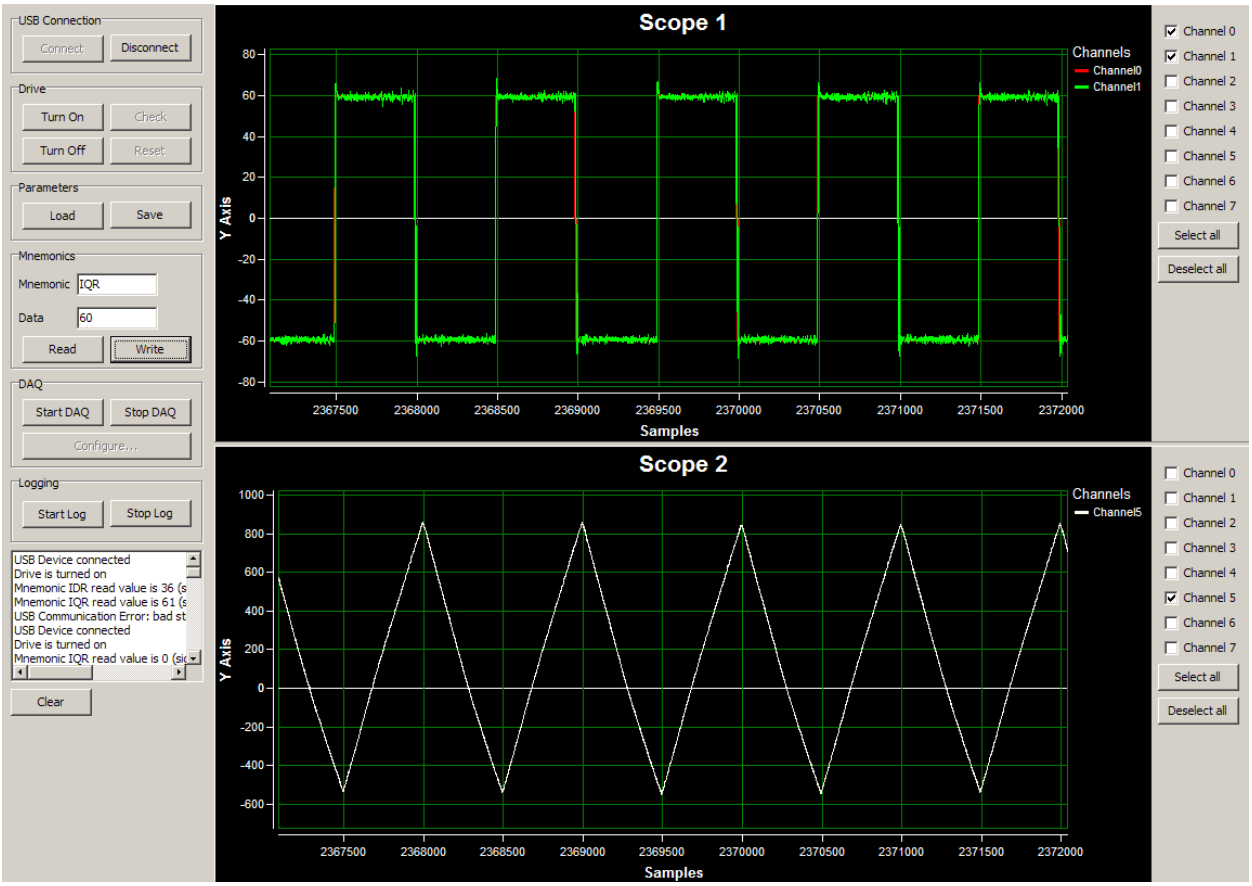
7.3.6. U okviru gornjeg osciloskopa selektovati kanale 0 i 1 a u okviru donjeg kanal 5 (Slika 7.1).



Slika 7.1 Izgled GUI prozora nakon selektovanih kanala

Time je omogućeno da se na gornjem osciloskopu prikazuju referentna i merena struja i_q a na donjem brzina obrtanja motora.

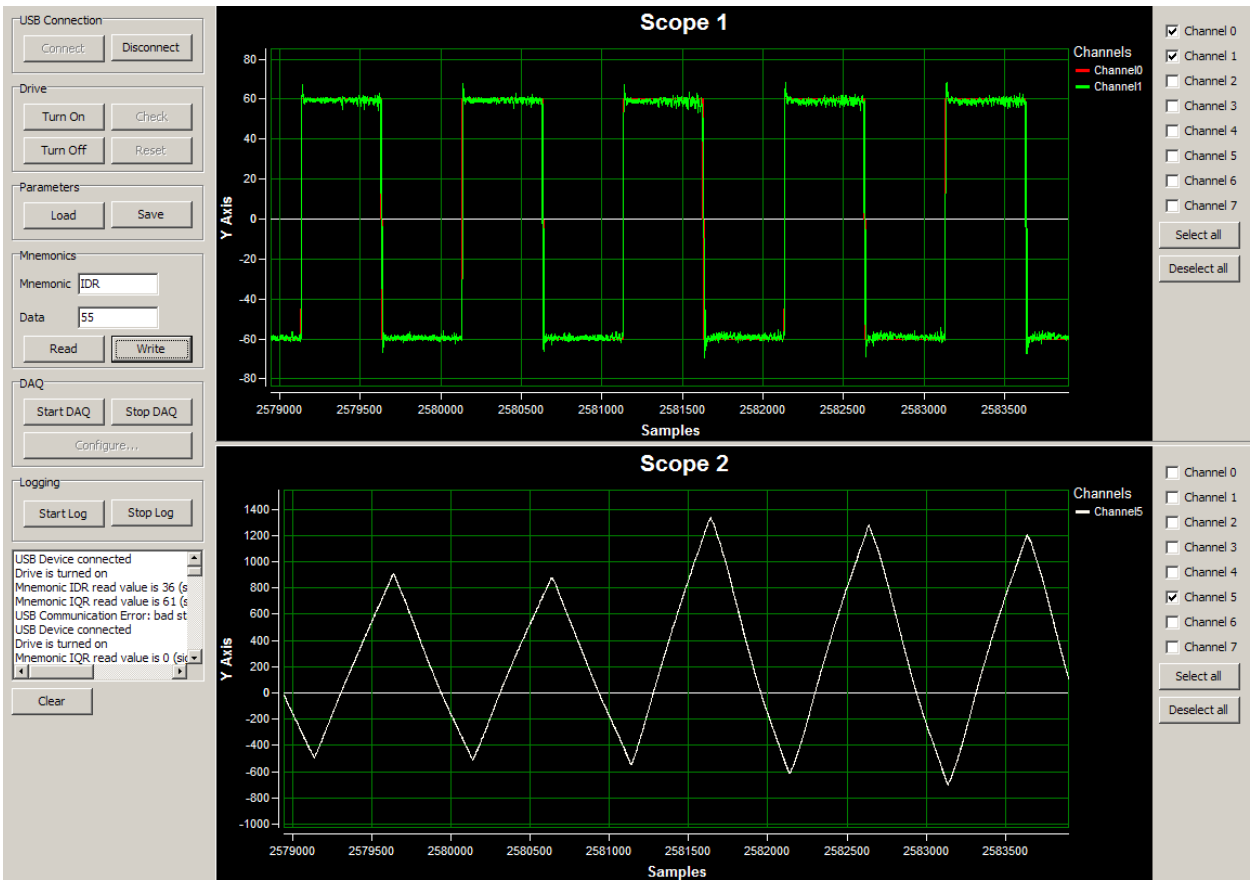
7.3.7. Postaviti struju i_q na 1/2 nominalne vrednost koja je upisana u Tabeli 7.1. Kao što je to bio slučaj u vežbi „Digitalna regulacija struje“, referenca u q osi će menjati svoj znak ali sada na svakih 200 msec. Time se obezbeđuje impulsna promena momenta motora. Izgled osciloskopa nakon postavljanja q struja na nominalnu vrednost bi trebalo da izgleda kao na slici 7.2. Pokrenuti logovanje podataka u dužini trajanja reda 1 sec. Procedura logovanja opisana je u uputstvu vežbe „Digitalna regulacija struje“ pod tačkom 4.4.2.7. Logovani podaci će se koristiti u tački 7.3.12 tako da za sada treba pratiti dalje uputstvo.



Slika 7.2 Prikaz referentne i merene struje q (gore) i kao i brzine (dole)

Moguće je uočiti da se brzina menja linearno (donji grafik na Slici 7.2) što je i opravdano imajući u vidu da je motor neopterećen i da je izvod brzine konstantan. Takođe, kako moment menja znak (može se reći da moment ima isti oblik kao i struja q) tako i brzina menja strminu.

7.3.8. Promeniti vrednost struje i_d na polovinu nominalne vrednosti. Izgled osciloskopa će biti nalik na sliku 18.



Slika 7.3 Prikaz referentne i merene struje *q* (gore) i kao i brzine (dole) prilikom prmene struje *d*

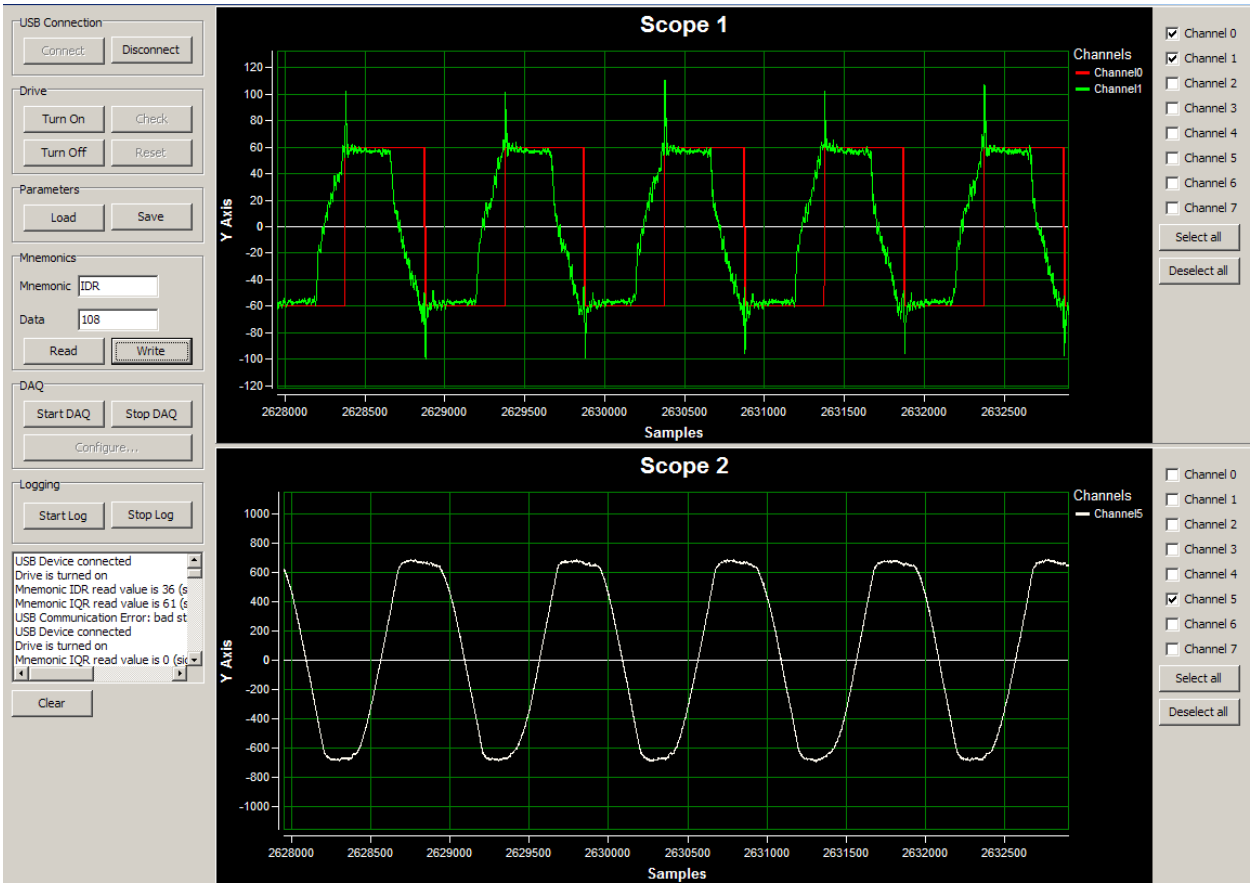
Na polovini talasnog oblika brzine je nastupila promena struja *d*. Jasno se primećuje da strmina brzine počinja da sepovećava što znači da se i momenat povećava. Ovim je pokazano da je momenat zapravo proizvod struja *d* i *q* ose.

7.3.9. Konstantu *slipgain* postaviti na nominalnu vrednost (pogledati Tabelu 1). Zatim i struju u *d* osi postaviti na nominalnu vrednost (Tabela 6.2).

7.3.10. Izgled ekrana nakon promena u tački 7.3.9. bi trebao da bude nalik na Sliku 7.4.

7.3.11. Na osnovu Slike 17 odrediti momenat motora. Za određivanje momenta potrebno je poznavati momenat inercije motora *J* koji iznosi 0.0011 kgm^2. Odgovor uneti u izveštaj koji se odnosi na ovu laboratorijsku vežbu i koji se nalazi u poglavlju 7.4.

7.3.12. Talasne oblike dobijene logovanjem u tački 7.3.7 treba iscrtati korišćenjem matlab fajla koji se nalazi u projektnom direktorijumu. Dobijene grafike uneti u izveštaj u poglavlju 7.4.



Slika 7.4 Prikaz stanja gde postoji odsustvo upravljanja

Na gornjem delu grafika se može uočiti da struja u *q* osi više ne uspeva da prati referencu kao i to da brzina više nije testerasta iz razloga što se ni moment ne menja impulsno. Razlog za to je nedostatak naponske margine i nemogućnost injektovanja viših vrednosti struja pri povišenim brzinama.

7.3.13. Po završetku rada vežbe zaustaviti pogon pritiskom na polje *Turn Off*. Time će se pogon zaustaviti i invertor neće obezbeđivati napajanje motora.

7.5 Izveštaj

Odgovori:

Student:

Ime: _____

Prezime: _____

Broj indeksa: _____

Laboratorijska postavka _____

7.3.11. Momenat motora iznosi _____

7.3.12.

Struja i_q i i_{qref} (USB osciloskop)

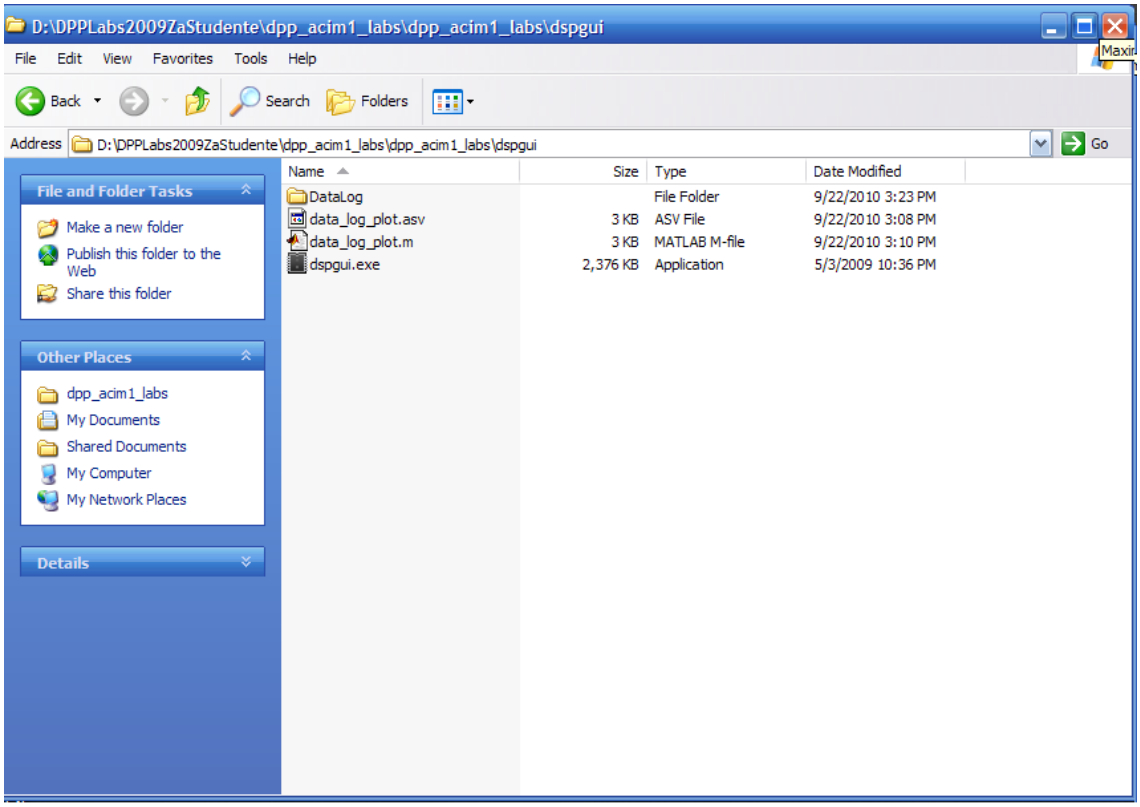
8. Dodatak A - Uputstvo za korišćenje USB osciloskopa

Podaci koje DSP prosleđuje računaru preko USB veze mogu se prikazivati na ekranu PC-a u realnom vremenu pomoću posebnog programa *dspgui.exe* koji zbog njegove uloge i sličnosti sa stvarnim osciloskopom nazivamo *USB osciloskop*. Ovaj dodatak sadrži detaljno uputstvo za korišćenje USB osciloskopa koji u našoj vežbi predstavlja osnovni alat za praćenje najznačajnijih veličina motora u realnom vremenu. Pre početka korišćenja USB osciloskopa, tj. ovog uputstva podrazumeva se da je ostvarena fizička konekcija USB kablom između digitalne ploče i RS-a. Detalji o načinu povezivanja digitalne ploče i RS-a mogu se pročitati u poglavlju 1 i/ili poglavlju 2.

Uputstvo je izdeljeno na sledećih osam koraka.

1. **Pronalaženje programa *dspgui.exe***

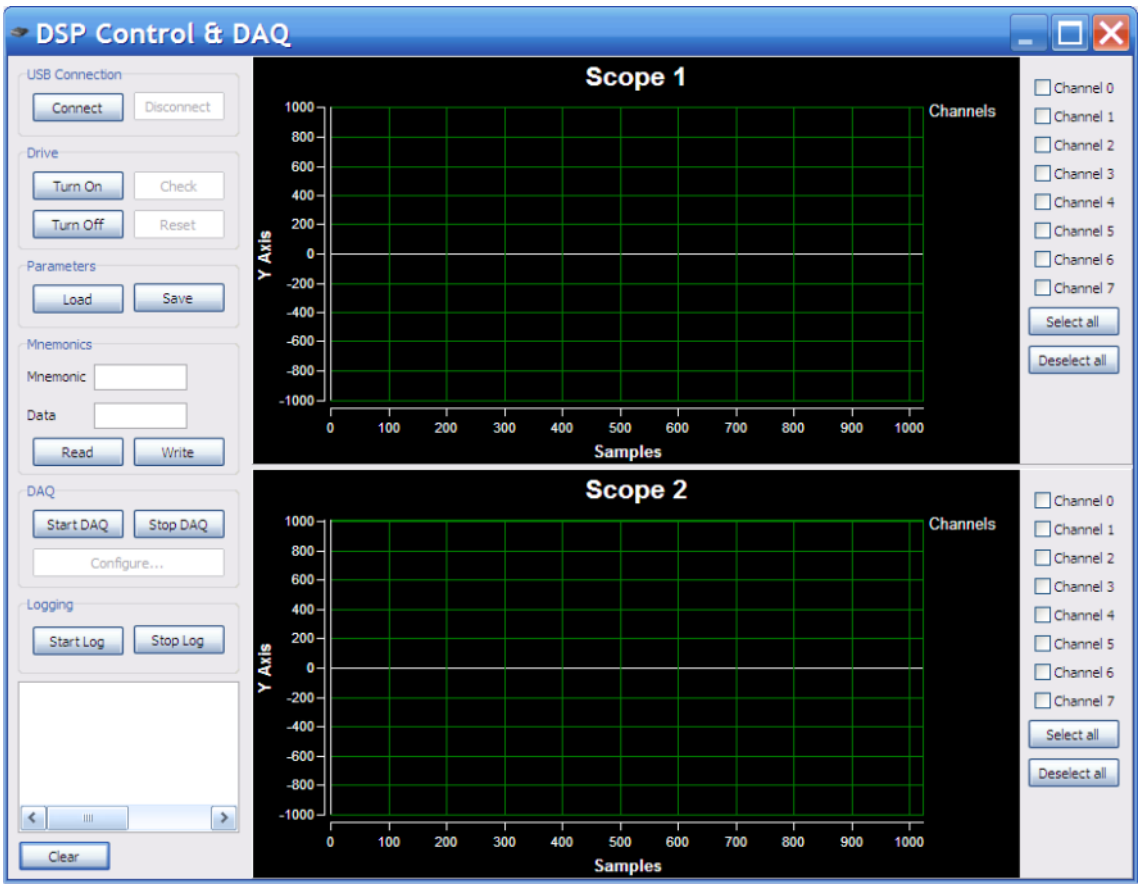
   U osnovnom folderu za konkretni deo vežbe locirati folder ***dspgui*** i otovoriti ga. U njemu se nalazi izvršni fajl ***dspgui.exe***. Sadržaj foldera ***dspgui*** prikazan je na Slika A1.



*Slika A1 – Izgled otvorenog foldera **dspgui***

2. **Pokretanje izvršnog fajla *dspgui.exe***

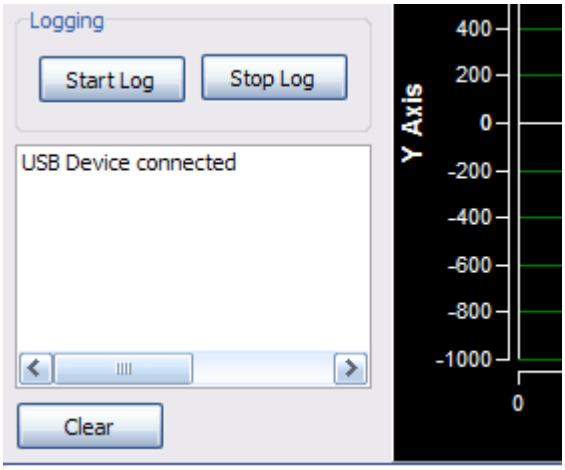
   Pokretanjem izvršnog fajla ***dspgui.exe*** otvara se prozor kao na Slika A2.



Slika A2 – Izgled grafičkog okruženja izvršnog programa

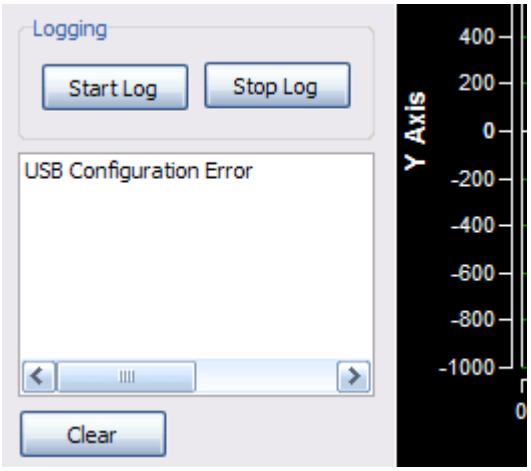
3. Ostvarivanje komunikacije između DSP-a i RS-a

Pritisnuti dugme *Connect*. Nakon uspešnog uspostavljanja komunikacije računara i DSP-ja u statusnom prozoru u donjem levom delu ekrana će se ispisati *USB Device Connected* kao što je prikazano na Slika A3.



Slika A3 – Izgled statusnog prozora pri uspešno uspostavljenoj komunikaciji

Ukoliko se komunikacija ne uspostavi, u statusnom prozoru će biti ispisano *USB Configuration Error* kao na Slika A4.



Slika A3 – Izgled statusnog prozora pri neuspešnom pokušaju ostvarivanja komunikacije

U ovom slučaju traba sačekati 10-20 sekundi pa pokušati ponovo –i to maksimalno tri puta. Ukoliko program i dalje odbija uspostavljanje veze, proveriti da li su sve hardverske komponente ispravno povezane i da li je uspostavljena fizička konekcija putem USB kabla. Ukoliko USB kabal nije povezan – uspostavljanje veze je definitivno nemoguće. U koliko se ustanovi da je sve ispravno povezano, a pogon ne radi, pozvati dežurnog asistenta/laboranta.

4. Akvizicija podataka

Pritisnuti dugme *Start DAQ* da bismo pokrenuli proces akvizicije odnosno prikupljanja podataka iz DSP-a(engl. ***D**ata **Acq**uisition - **DAQ***). Akvizija podataka će nam omogućiti da posmatramo veličine iz DSP-a na ekranu računara u realnom vremenu. Pritiskom na dugme *Stop DAQ* zaustavljamo ovaj proces. Oba grafika („ekrana osciloskopa“) *Scope1* i *Scope 2* mogu da prate 8 istih kanala sa DSP-a (zapravo osam promenljivih). Izbor kanala se vrši štikliranjem odgovarajućih *checkbox*-ova sa desne strane ekrana. *Sheckbox*-ove možemo zamisliti kao kanale realnog osciloskopa čijim odabirom prikazujemo odgovarajuću veličinu. U konkretnom delu vežbe biće napomenuto koje *checkbox*-ove treba štiklirati da bi se prikazale željene veličine iz DSP-a.

5. Startovanje pogona

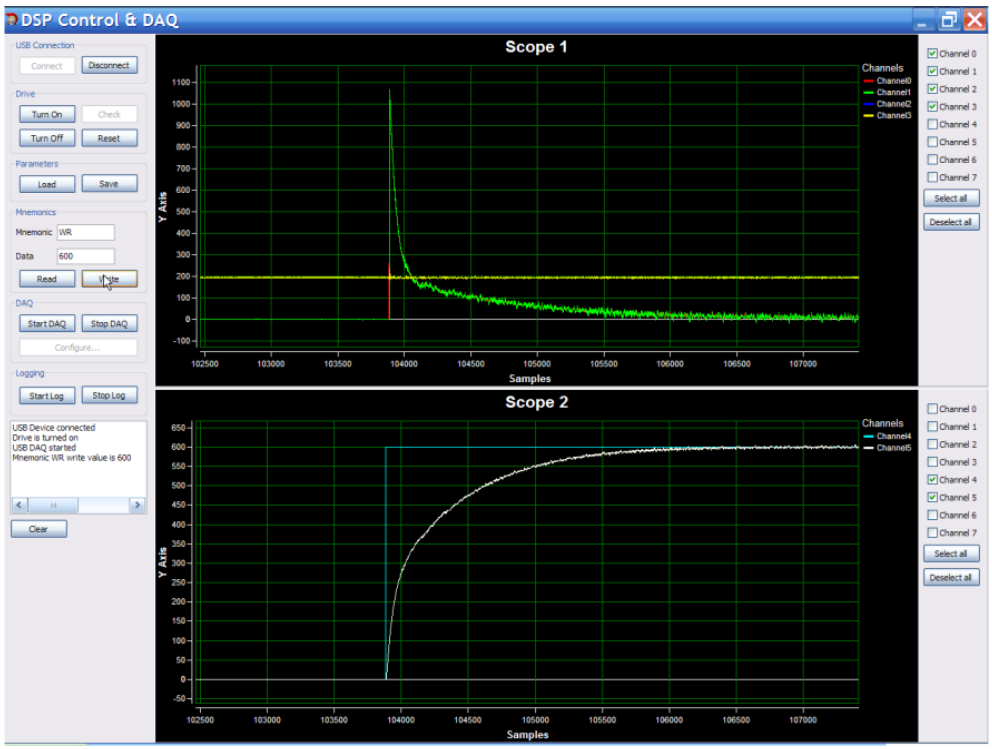
Pretpostavlja se da je pre startovanja pogona iz programa ***dspgui.exe*** student prošao kroz proceduru povezivanja i uključivanja hardvera koja je opisana u poglavljima 1 i 2.

Pogon startujemo pritiskom na dugme *Turn On*. Pritiskom na dugme *Turn Off* moćemo da gasimo pogon.

Po uključenju pogona trebalo bi da se uključi zelena LED na digitalnoj ploči. Ukoliko se ne upali zelena, a upali crvena LED to je indikacija greške. Jedan od uzroka ove situacije može biti taj što prekidač Power nije uključen. Po uključenju ovog prekidača nakon par sekundi crvena LED bi trebala da se isključi. Ukoliko se ne isključi, pozvati dežurnog asistenta/laboranta.

6. Prikaz veličina na ekranima *Scope1* i *Scope2*

Da bismo tokom rada pogona prikazali željene veličine, čekiramo odgovarajuće *checkbox*-ove. Primer prikaza nekih veličina pogona na ekranima USB osciloskopa (*Scope1* i *Scope2*) je dat na Slika A4. Na apscisi se nalazi broj odbirka, a na ordinati vrednost veličine koja se posmatra.

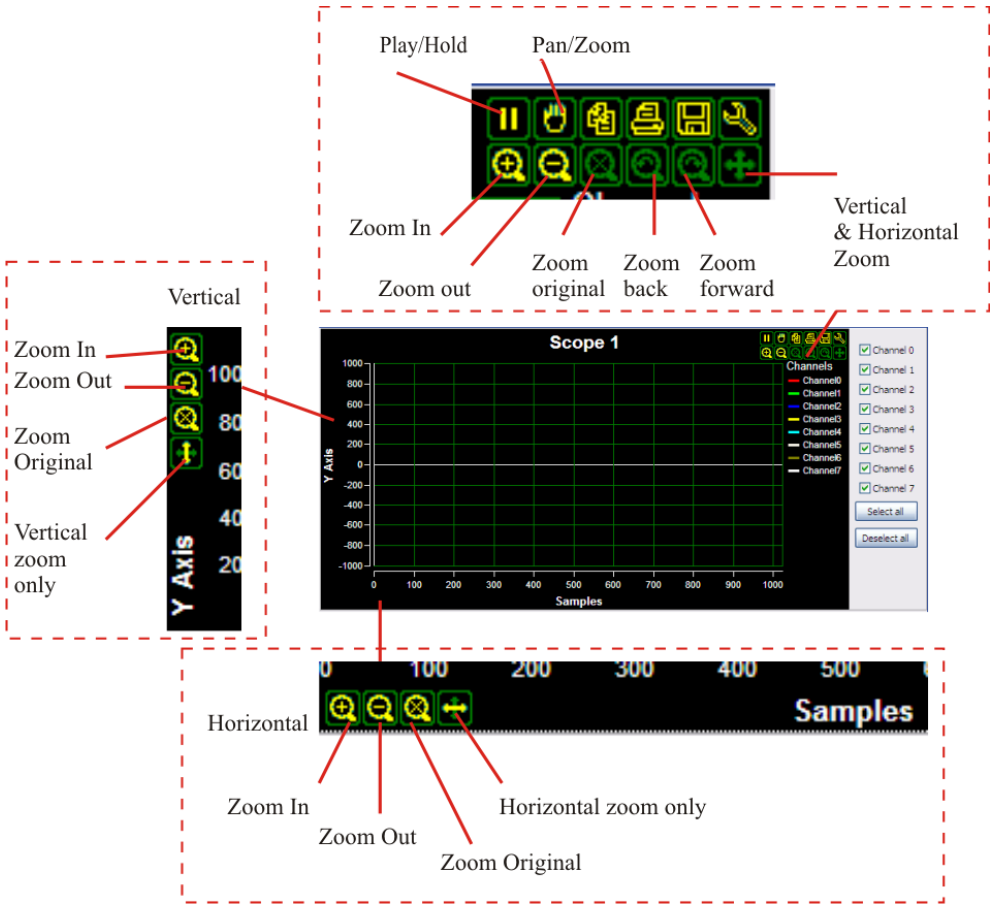


Slika A4 – Prikaz nekih veličina pogona na Scope1 i Scope2

7. Alati za podešavanje prikaza veličina na ekranima *Scope1* i *Scope2*

Kada dovedemo miš na jedan od grafika pojaviće se meni u gornjem desnom uglu, **Error! Reference source not found.**. Još dva manja menija se prikazuju kada se miš približi apscisi (*Samples*) i ordinati (*Y Axis*) i služe za upravljanje apscisom i ordinatom, respektivno.

- Dugme *Play/Stop*. Režim *Play* (prikazane dve horizontalne crtice) – omogućava se iscrtavanje grafika. Režim *Stop* (prikazan trougao) pauzira iscrtavanje. Uzastopnim pritiscima menjamo režim.
- Dugme *Pan/Zoom*. Režim *Pan* (prikazana lupa) omogućava pomeranje ekrana grafika gore dole, i ako je zumiran grafik levo desno. Režim *Zoom* (prikazana ručica)– omogućava zumiranje, po obe ose.
- Kontrolna dugmad za zumiranja na glavnom meniju: *Zoom In, Zoom Out, Zoom Original, Zoom Back, Zoom Forward i Vertical & Horizontal Zoom (Enable)* – govore same za sebe i tiču se zumiranja grafika. Ovo se odnosi na režim zumiranja, kada je na kontrolnom dugmetu Pan/Zoom prikazana ručica. Da bi se zumirale obe ose mora da se klikne na dugme *Vertical & Horizontal Zoom (Enable)*.
- Vertikalne kontrole. Omogućavaju kretanje/zumiranje samo po vertikalnoj osi.
- Horizontalne kontrole. Omogućavaju kretanje/zumiranje samo po horizontalnoj osi.



Slika A5. Kontrole grafičkog korisničkog interfejsa

- **VAŽNA NAPOMENA**. Ovi grafici su moćna alatka za vizuelnu analizu veličina. Međutim, ovi alati su malo kompleksniji i ne treba gubiti mnogo vremena u ovom trenutku da se potpuno shvati njegovo funkcionisanje. Potrebno je malo iskustva, koje će student steći tokom izrade vežbi.

8. **Pauziranje iscrtavanja signala na oba grafika.** Pritiskom na dugme **F1** Možemo da pauziramo iscrtavanje na oba grafika. Slično kao funkcija *Play/Stop*, s tim što je ova funkcija je većeg prioriteta i važi za oba grafika.

9. Dodatak B – Predstavljanje hardvera

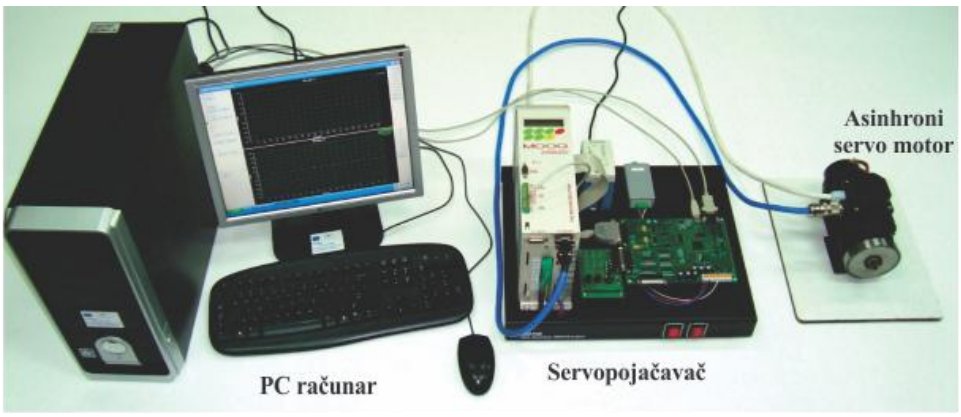
U narednom poglavlju je dat detaljniji prikaz opreme koja se koristi u MOT09 (TEMPUS) vežbi. U njemu se nalaze detaljnije informacije kako bi zainteresovani studenti imali priliku da dodatno prouče laboratorijske postavke. Za rad vežbama nije neophodno proučiti ovaj dokument. Neki delovi opreme su u određenoj meri proučeni u delovima vežbi, „Digitalna regulacija struje“ , „Indirektno vektorsko upravljanje“, kao i DSP periferije.

Glavni delovi postavki su predstavljeni u nekoliko odeljaka pri čemu su prikazane i slike najbitnijih delova opreme. Cilj je da se studenti upoznaju sa elementima i načinom povezivanja istih kako bi lakše pristupili praktičnom radu u laboratoriji.

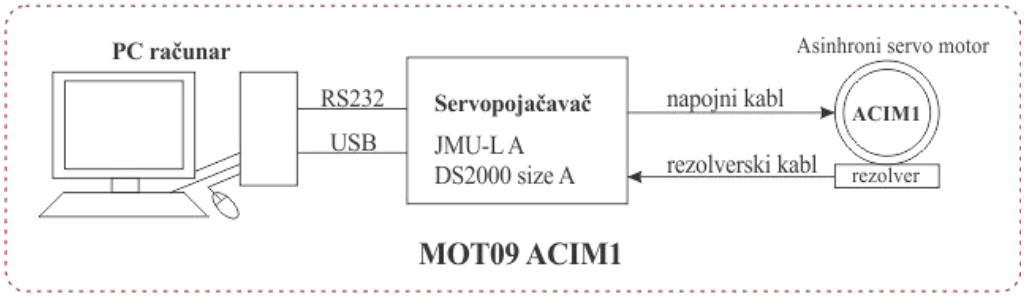
Osnovni hardverski delovi postavki

Za potrebe laboratorijskih vežbi u cilju napajanja motora koriste se invertori proizvođača *MOOG*, motori, eksterne DSP ploče (JMU-L ploče, projektovane u Laboratoriji za digitalno upravljanje na ETF Beograd) i PC računari. Korišćenjem ovih komponenti realizovane su 3 različite laboratorijske postavke, ACIM1, ACIM2 i PMSM. Slova u nazivu postavke označavaju vrstu motora koji se koristi u postavci (ACIM je vezano za *„alternating current induction machine“* - asinhroni motor, dok je PMSM *„permanent magnet synchronous machine“* vezano za sinhroni motor sa stalnim magnetima).

Korišćeni invertori su iz serije DS2000 i postoje 3 veličine, Size A, Size B i µDS. Motori koji se koriste u vežbi su različitih karakteristika i proizvođača, MOOG (asinhroni servomotor-crne boje), ABB (asinhroni motor-plave boje) i Vickers (sinhroni motor sa stalnim magnetima). Eksterne DSP ploče su iste za sve 3 postavke. Izgled, struktura, način povezivanja kao i karakteristike ovako realizovanih pogona prikazani su na slikama 1, 2 i 3.



a) Izgled postavke

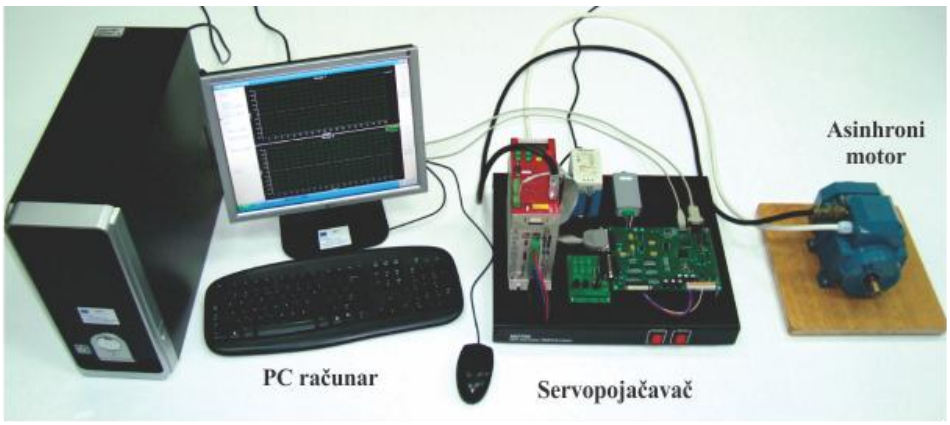


b) Blok dijagram

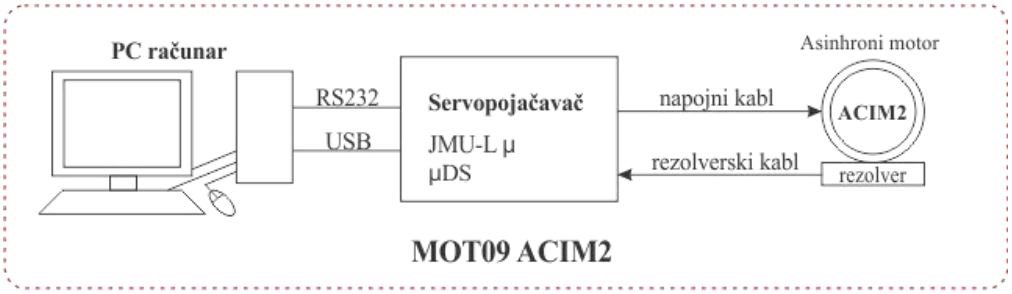
Motor ACIM1 - MOOG FAS Y063-V	
ωnom	3000 ob/min
$fnom$	100 Hz
Broj polova motora	4
$Unom$	325 V
$Inom$	2,1 A
Sprega statorskog namotaja motora	Y
$Mmax$	6,2 Nm
Senzor pozicije	rezolver
Broj polova rezolvera	4
MOOG DS2000 Size A	
$Iout\ max$	3 A
$Iout\ peak$	9A

c) Tehnički podaci

Slika 1 – Postavka ACIM1



a) Izgled postavke

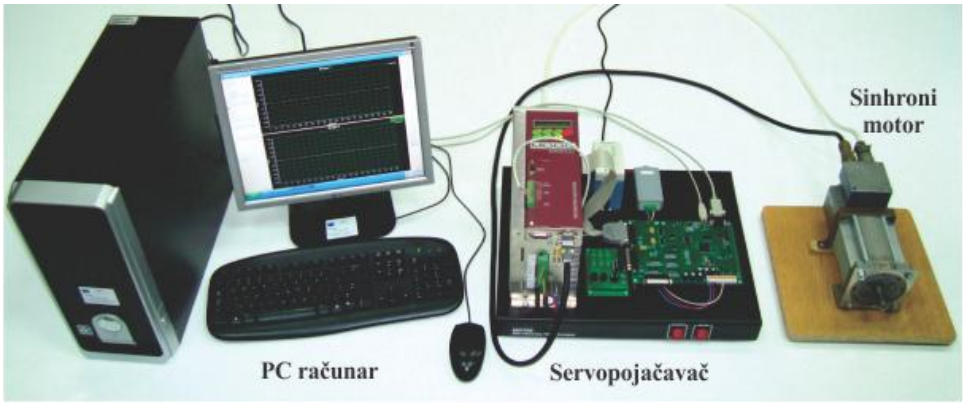


b) Blok dijagram

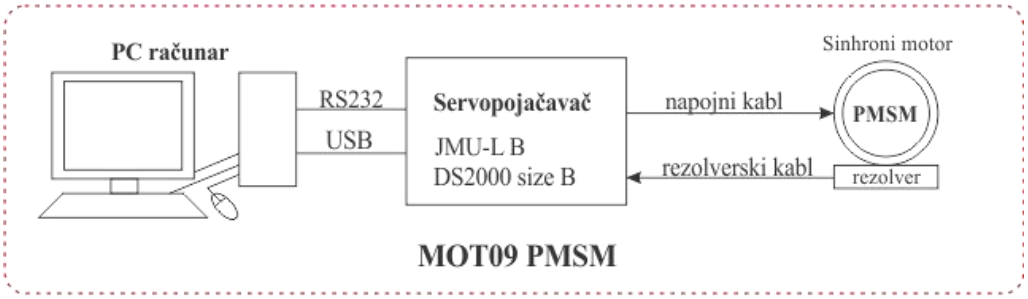
Motor ACIM2 - ABB M2VA71C4	
ωnom	1410 ob/min
$fnom$	50 Hz
Broj polova motora	4
Unom	380 V
Inom	1,45 A
Sprega statorskog namotaja motora	Y
Pnom	0,55 kW
Mmax	3,74 Nm
Senzor pozicije	rezolver
Broj polova rezolvera	2
MOOG DS2000 µDS	
Iout max	6 A
Iout peak	22 A

c) Tehnički podaci

Slika 2 – Postavka ACIM2



a) Izgled postavke



b) Blok dijagram

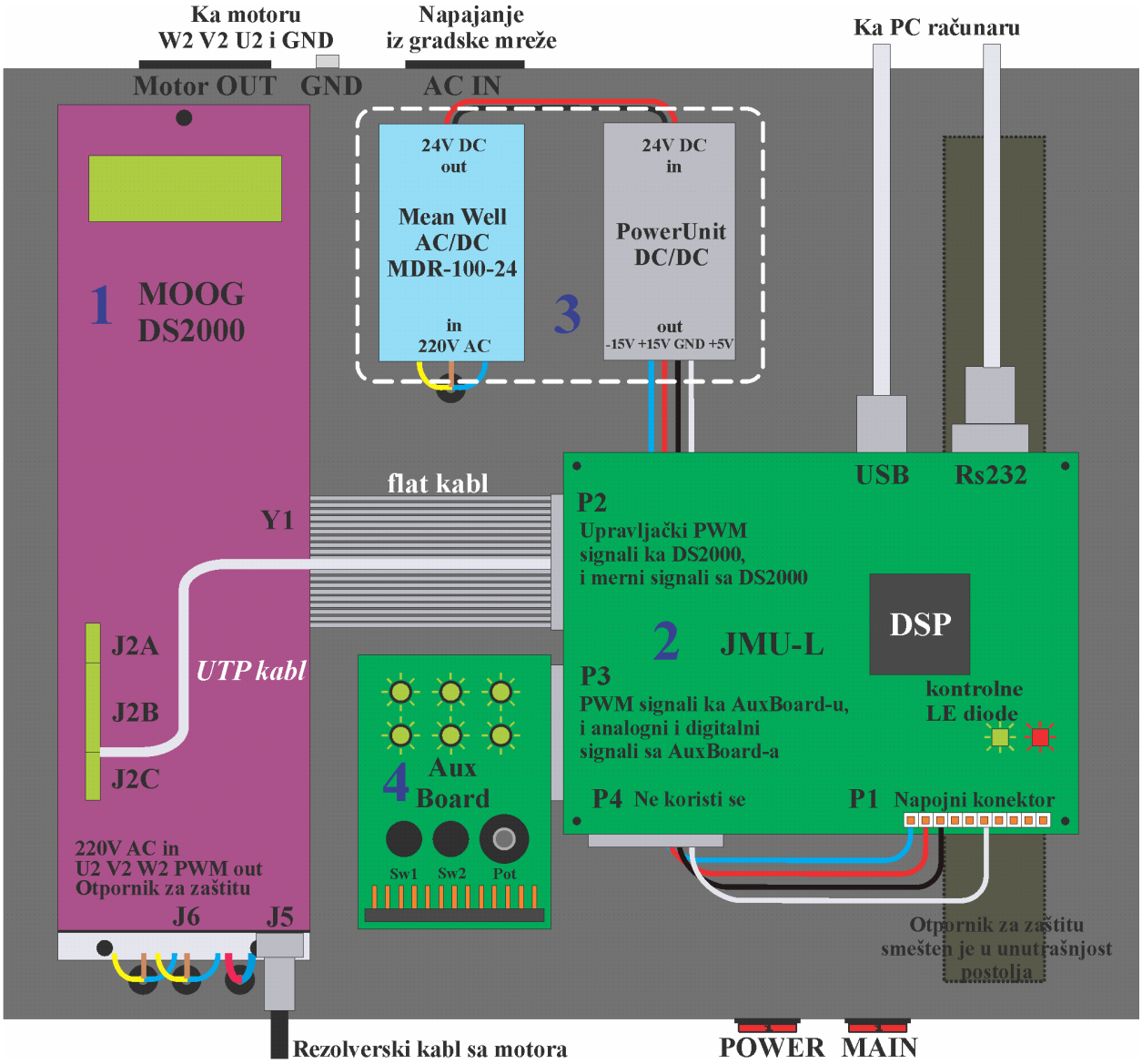
Motor PMSM - Vickers FAS T1 M4 030	
ωnom	3000 ob/min
$fnom$	150 Hz
Broj polova motora	6
$Unom$	180 V
$Inom$	4,8 A
Sprega statorskog namotaja motora	Y
$Pnom$	0,97 kW
$Mmax$	12,5 Nm
Senzor pozicije	rezolver
Broj polova rezolvera	6
MOOG DS2000 Size B	
$Iout\ max$	14 A
$Iout\ peak$	42A

c) Tehnički podaci

Slika 3– Postavka PMSM

Korišćeni invertori imaju mogućnost da samostalno upravljaju motorima. Naime, svaki od invertora poseduje svoju upravljačku jedinicu u kojoj se nezavisno izvršava kompletan algoritam upravljanja, putem Digitalnog Signal Procesora (DSP) sa odgovarajućim periferijama. Međutim za potrebe laboratorijske vežbe upravljanje je izmešteno iz DSP-a, koji se nalazi u sklopu invertora, u DSP TMS320LF2407A koji se, sa svojim pratećim perifernim komponentama, nalazi na JMU-L elektronskoj ploči. Praktično, DSP sa JMU-L ploče tako preuzima ulogu regulatora u svakoj postavci.

Kako je kontrola motora izmeštena van invertora nadalje će se pod pojmom *servopojačavač* smatrati sklop koji je sačinjen od invertora i eksterne DSP ploče. Bitno je istaći da se signali za paljenje tranzistora (prekidača u invertoru), šalju sa DSP pločice *flat* kablom do upaljača koji se nalaze unutar invertora. Na taj način je ostvarena veza između energetskog modula invertora i upravljačkog bloka servopojačavača. Blok šema servopojačavača data je na slici 4.



Slika 4 – Blok šema servopojačavača

Na slici 4 je moguće uočiti 4 bloka:

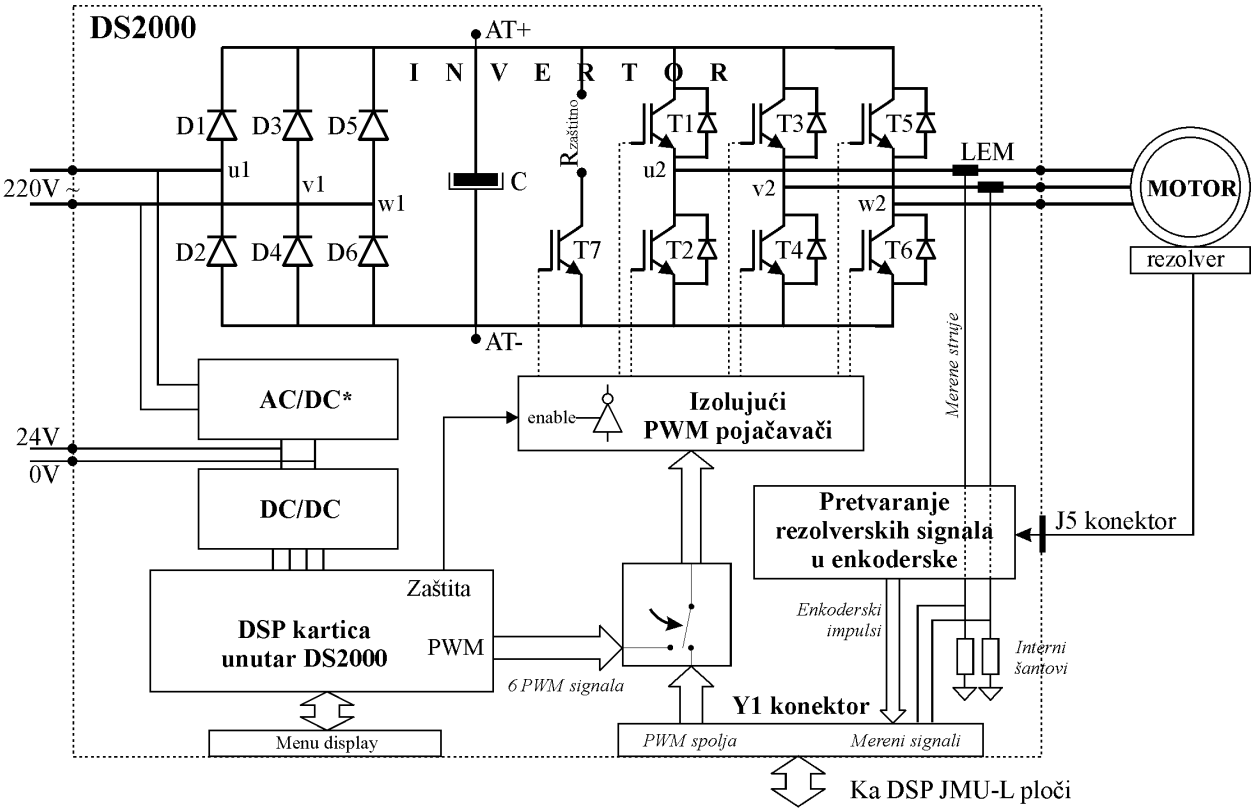
1. Invertor (DS2000 Size A, Size B i µDS)
2. Eksterna DSP ploča (JMU-L)
3. Blok za napajanje; AC/DC napajanje 220V AC-24V DC (Mean Well MDR 100-24) i DC/DC (24V DC +/-15V(DC), +5V(DC) i GND)
4. Pomoćna pločica Aux-Board
5. Otpornik za kočenje.

U narednom odeljku će biti nešto detaljnije opisana funkcija svih navedenih blokova.

Invertor DS2000

Kao što je već pomenuto, invertor ima mogućnost da samostalno upravlja motornim pogonom. Poseduje energetske i upravljačke blokove neophodne za kontrolu motora, kao i odgovarajući interfejs za komunikaciju sa korisnikom (*menu* – displej) putem koga je moguće čitanje, unos i izmena određenih parametara za kontrolu pogona. U laboratorijskoj vežbi je kontrola motora izmeštena u DSP JMU-L ploče o čemu je već bilo reči u prethodnom odeljku. Međutim zadržane su određene funkcije DSP-a koji se nalazi unutar invertora kao što su prekostrujna, prenaponska i temperaturna zaštita, pretvaranje rezolverskih signala u enkoderske impulse itd.

Na Slici 5 prikazana je blok šema invertora.



● Ovi signali se vode na napojni J6 конектор

* Uređaj μDS ne sadrži ovaj blok, pa se zato na njega mora eksterno dovoditi napon od 24V

$R_{zaštitno}$ Otpornik za zaštitu od prenapona se spolja dodaje, i nalazi se unutar kutije servopojačavača

Slika 5 – Blok šema invertora DS2000

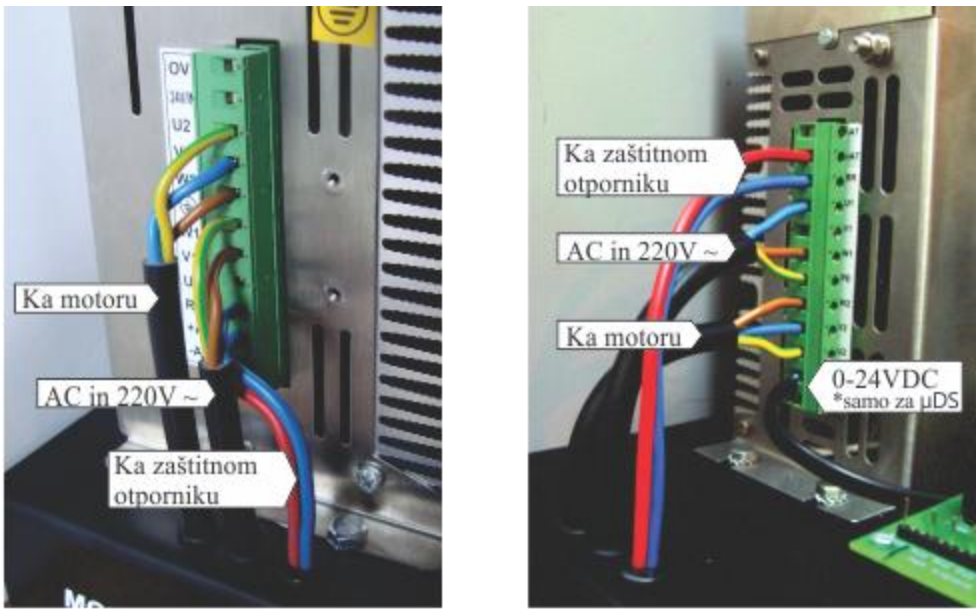
Na slici 5 je moguće uočiti nekoliko blokova. Energetski blok čine: ulazni stepen invertora – trofazni ispravljač; jednosmerno međukolo u kome se nalazi elektrolitski kondenzator priključen na + i – kraj jednosmernog međukola i otpornik za kočenje sa redno priključenim tranzistorom. Izlazna sekcija invertora koju čine prekidački elementi (IGBT tranzistori) koji formiraju trofazni „*H*" most takođe je deo energetskog bloka. Pored energetskog bloka, može se uočiti i blok koji se u širem smislu može nazvati modul za obradu signala, koga čine DSP koji ne vrši upravljanje, AD konvertor kao i deo za pretvaranje rezoverskih signala u enkoderske impulse. Sastavni deo svakog regulisanog pogona je merni sistem i u navedenim laboratorijskim vežbama on uključuje merenje struje, napona jednosmernog međukola kao i merenje brzine.

Povezivanje invertora i DSP ploče kao i invertora i odgovarajućih naponskih izvora je ostvareno putem nekoliko konektora. U nastavku je prikazan kratak opis konektora koji se koriste u ovoj laboratorijskoj vežbi odnosno njihovih primarnih funkcija (detaljniji opis svih raspoloživih konektora se može naći na sajtu proizvođača http://www.MOOG.com).

Konektor J6 ima 12 pinova i omogućava povezivanje invertora sa**:**

- Ulaznim naizmeničnim trofaznim napajanjem, krajevi U1,V1 i W1 (u laboratorijskoj vežbi invertor je povezan na monofazno napajanje od 220 V preko krajeva U1 i W1);
- Motorom, krajevi U2, V2 i W2;
- Zaštitnim otpornikom (otpornikom za kočenje) koji se nalazi u jednosmernom međukolu;
- Jednosmernim naponom napajanja od 24V; krajevi 0V i 24V, kod invertora Size A i Size B ostaju nepovezani jer postoji unutrašnji AC/DC pretvarač dok su kod μDS iskorišćeni usled odsustva AC/DC konektora unutar uređaja što se i vidi na Slici 5;

Izgled povezanog konektora u slučaju Size A/B kao i u slučaju μDS uređaja dat je na slici 6.



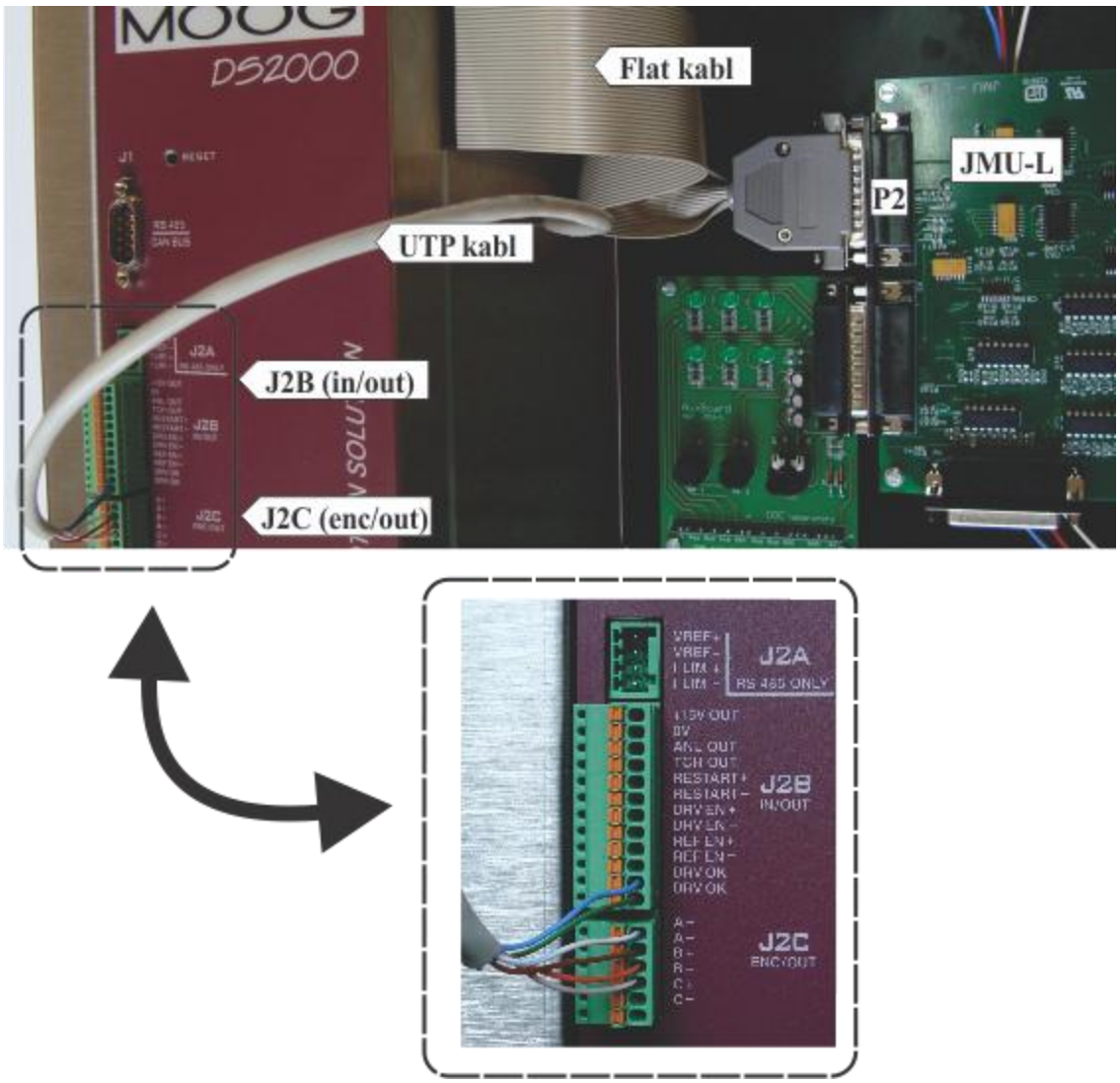
a) J6 - DS2000 Size A i DS2000 Size B b) J6 - µDS

Slika 6 – Izgled povezanog konektora J6

Konektori J2C (ENC/OUT) i J2B (IN/OUT) omogućavaju pristup digitalnim ulazima i izlazima od kojih se koriste:

- *Drive ok* (sa konektora J2B), digitalni signal koji se putem *flat* kabla vodi preko porta P2 do JMU-L DSP-a i daje informaciju o ispravnosti invertora, čije se stanje može odrediti na osnovu rada kontrolnih LE dioda (Slika 4);
- Signali A+, A-, B+ i B- (sa konektora J2C) koji predstavljaju emulirane enkoderske signale, dobijene na osnovu rezolverskih signala, koji se putem *flat* kabla vode preko porta P2 do jedinice za brojanje enkoderskih impulsa (QEP – Quadrature Encoder Pulses) u okviru DSP-a, o čemu se govori u vežbi „Indirektno vektorsko upravljanje“.

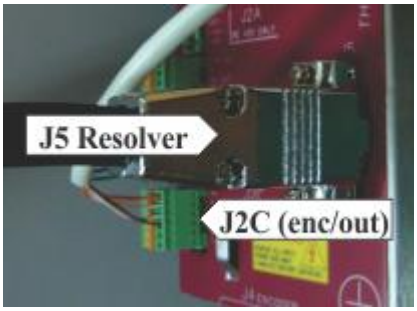
Potrebno je istaći da su pomenuti signali koji se koriste sa konektora J2B i J2C formirani zahvaljujući radu internog DSP-a (u sklopu invertora) koji je zadržao pomenute funkcije. Na sledećoj slici će biti predstavljena veza između invertora i JMU-L ploče.



Slika 7 - Veza između invertora i JMU-L ploče putem konektora J2B i J2C

Konektor J5 poseduje 9 pinova i služi za prijem signala sa rezolvera.

Na sledećoj slici je prikazan izgled konektora na koji je priključen kabl sa rezolvera. Takođe, ukazano je na konektor J2C koji daje enkoderske signale koristeći signale dobijene preko priključenog rezolvera.



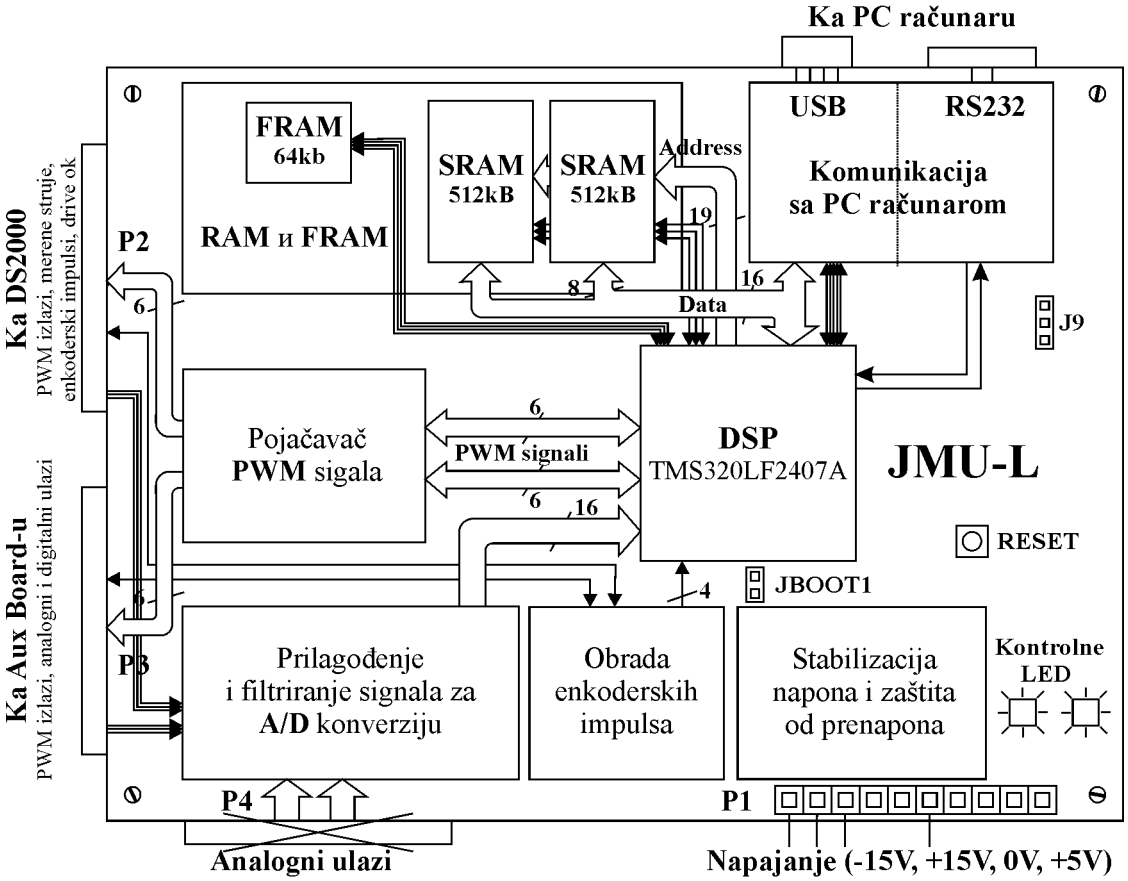
Slika 8 – Izgled konektora J5 sa priključenim rezolverskim kablom

Korisnički interfejs invertora neće biti obrađen iz razloga što se u laboratorijskoj vežbi ne koristi.

Eksterna DSP ploča (JMU-L)

Ploča je sačinjena od elektronskih komponenti koje čine upravljačku jedinicu invertora. Na ploči se mogu uočiti procesor i odgovarajuće periferijske komponente koje su sastavni deo u procesu upravljanja

motorom. Na slici 9 je prikazana principijelna šema JMU-L ploče gde su komponente predstavljene na način da odgovaraju stvarnom položaju na ploči.



Port P2 se koristi za upravljanje motorom
Port P3 se koristi za evaluaciju DSP-a
Port P4 se ne koristi u vežbi

Slika 9 - Šema JMU-L ploče

Od interesa je uočiti da signali o merenim veličinama dolaze do procesora gde se izvršava algoritam upravljanja pa se zatim preko pojačavača PWM signala šalju impulsi za paljenje tranzistora putem flat kabla na upaljače koji se nalaze u invertoru. Za komunikaciju sa PC računarom se koriste dva porta, USB (Universal Serial Bus) i serijski port (RS232) koji su na slici prikazani u gornjem desnom uglu. Serijska veza se koristi za upisivanje programskog koda u memoriju DSP-a dok se USB veza koristi za komunikaciju sa USB osciloskopom u realnom vremenu. Više o korišćenu USB oscilokopa može se pročitati u dodatku A pod nazivom Uputstvo za korišćenje USB osciloskopa.

Na ploči treba uočiti taster RESET koji resetuje DSP kao i džamper (JBOOT1) koji omogućava upisivanje programskog koda u DSP. Više o ovim elementima kao i o proceduri upisivanja koda u DSP može se pročitati u poglavlju Demo vežba, potpoglavlje Demonstracija U/f upravljanja (korak 3). Ostali delovi JMU-L ploče nisu toliko bitni za izvođenje laboratorijske vežbe pa iz tog razloga neće biti detaljnije opisivani.

Blok za napajanje

Blok za napajanje ima zadatak da obezbedi potreban napon za napajanje JMU-L ploče kao i invertora. Naponi koji su obezbeđeni ovim blokom su redom -15, 0, 5, 15, 24 V.

Pomoćna ploča AuxBoard

Pločica AuxBoard koju koristimo za manifestaciju rada periferija DSP-a prikazana je na slici 10 i na njoj su označeni važniji delovi.

Na njoj se nalazi 6 LE dioda, 2 tastera, 1 potenciometar, 1 analogni ulaz i letva sa mernim mestima.

LE (*Light Emiting*) diode D1 – D6. Koriste se da vizuelno dočaraju principe impulsno-širinske modulacije (PWM). Diode se takođe koriste i za vizuelnu verifikaciju pri evaluaciji digitalnih izlaza.

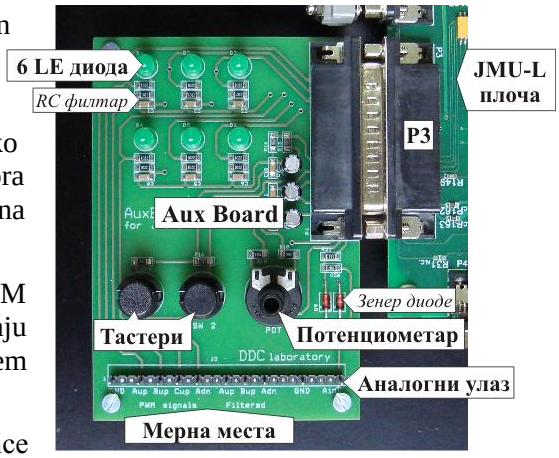
Tasteri SW1 i SW2. Služe za testiranje digitalnih ulaza DSP-a.

Potenciometar. Služi za zadavanje analognog napona. Napon se vodi na A/D konvertor DSP-a preko odgovarajućeg prilagodnog kola.

Analogni ulaz (Ain). Ovaj analogni ulaz se preko odgovarajućeg prilagodnog kola dovodi na ulaz A/D konvertora DSP-a. Napon na ulazu je ograničen zener diodama prikazanim na slici 2.1.

Merna mesta. Na njima se mogu posmatrati odgovarajući PWM signali i njihovi filtrirani ekvivalenti. PWM signali se filtriraju niskopropusnim *RC* filtrima, koji su prikazani na Slici 10 u gornjem levom uglu.

P3. Port koji predstavlja vezu između AuxBoard pločice

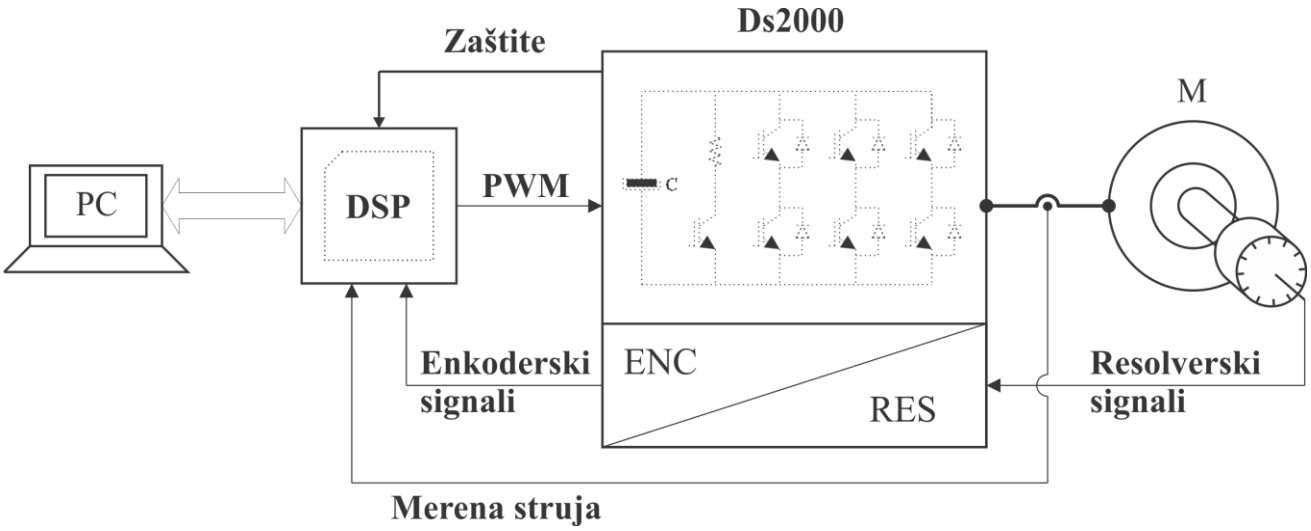


Slika 10. Izgled pomoćne pločice i digitalne JMU-L ploče.Aux Board

Rezime:

U laboratorijskim postavkama koristi se invertor, eksterna DSP ploča kao i motor sa svojim senzorima. Princip rada pogona je takav da se algoritam upravljanja izvršava na eksternoj DSP ploči, rezultat rada algoritma su vremena vođenja tranzistora, t_{on}, koja se šalju invertoru putem *flat* kabla. Na osnovu toga invertor dovodi širinski modulisan napon na priključne krajeve motora, a stanja motora (struje i brzina) se sa senzora vode do DSP ploče i time se zatvara povratna sprega.

Na slici 11 je prikazana principijalna šema celokupnog servopojačavača sa dodatkom PC računara i motora.



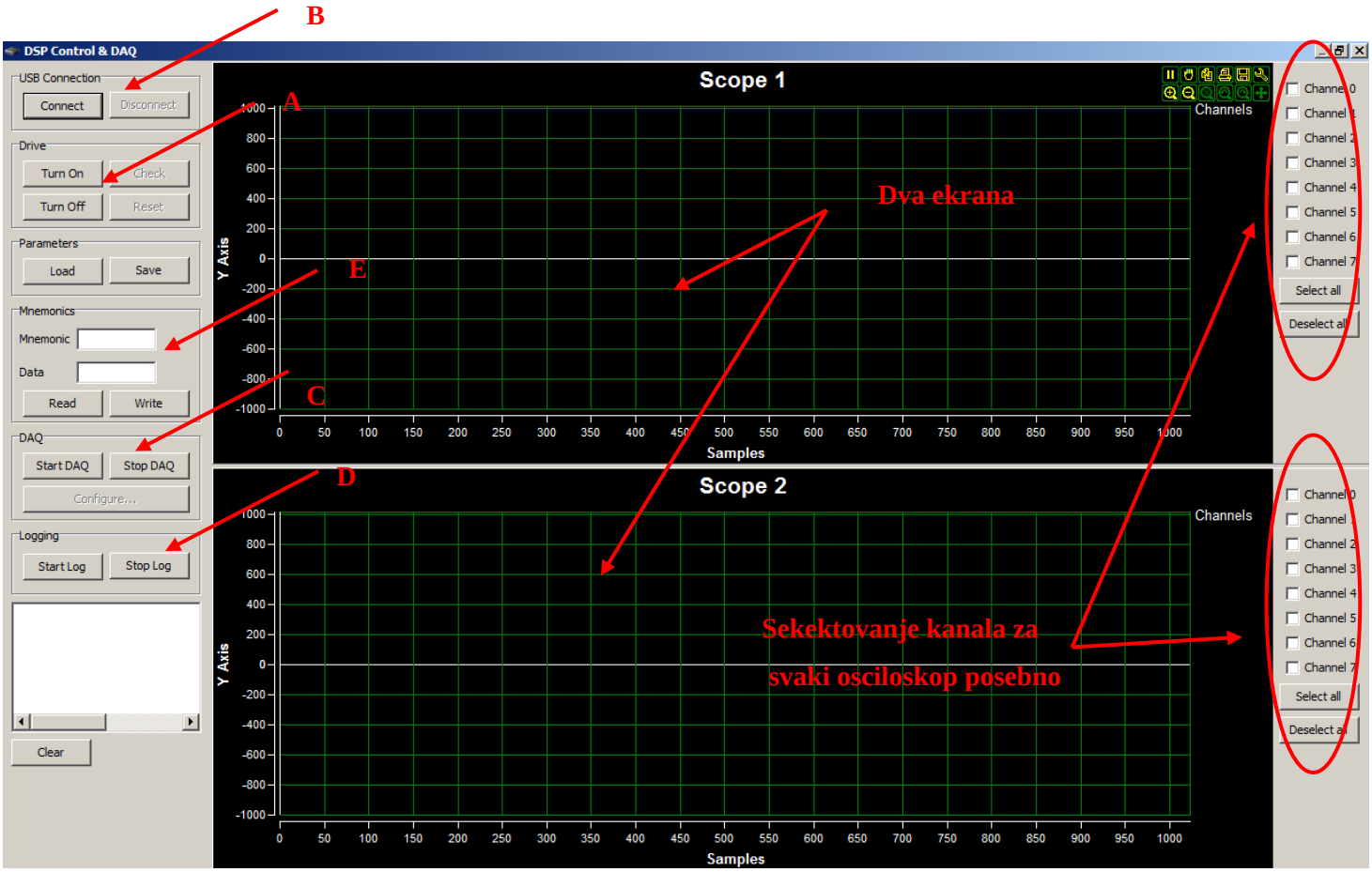
Slika 11 – Principijelna šema servopojačavača

Tok signala, komunikacija između pojedih delovaje takođe prikazana na slici 11.

PC računar

Povezivanjem PC računara sa DSP pločom uz pomoć DSP GUI-a omogućeno je praćenje određenih veličina u realnom vremenu, zadavanje referentnih vrednosti kao i analiza rada pogona poređenjem različitih

signala prikazanih na ekranu računara. Izgled ekrana putem kog se vrši komunikacija PC računara i DSP-a prikazana je na slici 12.



Slika 12 – Izgled ekrana USB osciloskopa

Na grafičkom prozoru treba razlikovati nekoliko različitih celina. Prvenstveno je moguće uočiti dva crna polja nalik na osciloskop, centralno postavljena. U tim poljima se prikazuju veličine koji korisnik definiše. Sa leve strane (slika 11) nalaze dugmići za:

- Aktiviranje pogona (detalj A)
- Uspostavljanje USB konekcije (detalj B)
- Uspostavljanje akvizicije podataka (detalj C)
- Rad sa logovanjem podataka prikazanih na osciloskopu (detalj D)
- Upis i čitanje raznih parametara pogona (detalj E).

Sa desne strane (Slika 11) nalaze se polja koja u slučaju selektovanja omogućavaju prikaz različitih različitih signala na osciloskopu.

Detalji vezani za karakteristike i način korišćenja ekrana GUI-a putem kog se vrši komunikacija PC računara i DSP-a dati su u dodatku A pod nazivom Uputstvo za korišćenje USB osciloskopa.