LABORATORIJA ZA MIKROPROCESORSKO UPRAVLJANJE ELEKTROMOTORNIM POGONIMA

LABORATORIJA ZA ELEKTRIČNU VUČU

Naziv vežbe

Laboratorijska stanica za ispitivanje algoritama upravljanja indirektno vektorski upravljanim asinhronim motorom

VEKTRA

Beograd, 2006.

SADRŽAJ

1. Uvod

Vežba je namenjena studentima koji pohađaju nastavu iz predmeta:

- Mikroprocesorsko upravljanje elektromotornim pogonima (TE5MUE)
- Električna vuča (EG4EV)

Cilj vežbe: Upoznavanje studenata sa savremenim digitalno regulisanim servo pogonom koji za izvršni organ koristi indirektno vektorski upravljani asinhroni motor. U tu svrhu, predviđeno je da studenti samostalno vežbaju na laboratorijskoj stanici VEKTRA, i kroz niz od 10 zadataka, praktično se upoznaju sa:

- H/W resursima savremenog servo pogona
  - Digitalnim kontrolerom
  - D/A konvertorima
  - Strujno regulisanim naponskim invertorom
  - Asinhronim motorom
  - Davačem (inkrementalnim optičkim enkoderom)
  - Brojačem enkoderskih impulsa
- S/W resursima savremenog servo pogona
  - *Real-time* prekidnom rutinom
  - Algoritmom za čitanje inkrementalnog enkodera
  - Algoritmom za generisanje obrtnih vektora
  - Algoritmom indirektnog vektorskog upravljanja
  - Digitalnim regulatorom brzine

Laboratorijska stanica VEKTRA se nalazi u Laboratoriji za Mikroprocesorsko upravljanje elektromotornim pogonima (ETF) u sobi 40a. Termini za vežbanje se određuju u dogovoru sa predmetnim asistentom. Za detaljne informacije obratiti se na:

lokal: 373 (soba 40a), ili 369 (soba 27)

ili na:

binella@ etf.bg.ac.yu

Sadržaj: Vežba je organizovana u sedam poglavlja sledećeg sadržaja:

Uvodne napomene date su u prvom poglavlju.

U drugom poglavlju, data su pitanja za proveru znanja.

U trećem poglavlju, naznačen je zadatak vežbe.

U četvrtom poglavlju, dati su teorijski osnovi neophodni za razumevanje i uspešnu realizaciju vežbe. Izloženi su koncepti indirektnog vektorskog upravljanja asinhronim motorom i nelinearne strujne regulacije.

U petom poglavlju, detaljno su opisani H/W i S/W resursi laboratorijske stanice VEKTRA, pocedura uključenja/isključenja laboratorijske stanice VEKTRA i postupak merenja izlaznih veličina laboratorijske stanice VEKTRA.

U šestom poglavlju, date su električne šeme veza H/W i izvorni C kod S/W laboratorijske stanice VEKTRA. Opisani su *Simulink* model brzinske regulacione petlje *speed.mdl* i program za indirektno vektorsko upravljanje asinhronim motorom sa grafičkim interfejsom *vektor.exe*.

Spisak referentne literature dat je u sedmom poglavlju.

Postupak:

Pre dolaska na vežbu:
- Pročitati tekst vežbe
- Odgovoriti na pitanja za proveru znanja (vidi Poglavlje 2)
- Odgovore uneti u Izveštaj
- Rešiti zadatke iz Poglavlja 3.3
- Rešenja zadataka uneti u Izveštaj

Po dolasku na vežbu:
- Rešiti zadatake iz Poglavlja 3.1 i 3.2
- Verifikovati rešenja zadataka iz Poglavlja 3.3

Po završetku vežbanja:
- Rešenja zadataka prikazati u Izveštaju
- Izveštaj predati predmetnom asistentu

Potrebni alati (samo za studente TE5MUE):

- PC računar
- ANSI C kompajler za DOS
- Programski paket *Matlab*/*Simulink* (verzija 5.1 ili više)

Potrebne datoteke (samo za studente TE5MUE):

- *Simulink* model *speed.mdl* i *Matlab* komandna datoteka *go_speed.m*
- Izvorni kod S/W laboratorijske stanice VEKTRA *source.zip*
- Programi *demo501.exe* i *demo126.exe*

Pomenute datoteke su dostupne na WEB *site*-u:

ddc.etf.bg.ac.yu

2. Pitanja za proveru znanja

U ovom poglavlju, data su pitanja za proveru znanja na koja treba odgovoriti pre dolaska na vežbu. Pitanja se odnose na laboratorijsku stanicu VEKTRA. Odgovore na pitanja uneti u Izveštaj.

Pitanje 1:
Za pogon lifta koristi se indirektno vektorski upravljani asinhroni motor sa podacima:

f_{nom} = 50 Hz, n_{nom} = 1400 ob/min, I_n = 20 A.

Ako je pobuda asinhronog motora nominalna ($I_d = I_{dnom}$), skicirati rukom talasni oblik jedne faze statorske struje asinhronog motora za slučaj kada je:

a) n = 0 [ob/min], $M_{opt} = M_{nom}$
b) n = 1400 [ob/min], $M_{opt} = M_{nom}$

Na svakom od talasnih oblika naznačiti kolika je amplituda i učestanost statorske struje.

Pitanje 2:
Koliko energetskih prekidača ima u invertoru laboratorijske stanice VEKTRA?

Pitanje 3:
Čemu služi otpornik za kočenje i kada se on koristi?

Pitanje 4:
Koliko iznosi mrtvo vreme (vreme potrebno za komutaciju dva energetska prekidača u istoj grani invertora) i na koji način je ono realizovano u laboratorijskoj stanici VEKTRA.

Pitanje 5:
Navesti pozitivna i negativna svojstva koncepta nelinearne strujne regulacije.

Pitanje 6:
Izlazni napon osmobitnog D/A konvertora uzima vrednosti iz opega [-10 ÷ 10] VDC. Ako se na ulaz D/A konvertora dovede broj 0040h, koliki napon će se pojaviti na njegovom izlazu?

Pitanje 7:
Inkrementalni optički enkoder sa 1000 markera i dva detekciona kompleta koristi se kao davač u laboratorijskoj stanici VEKTRA. Ako je brzina obrtanja vratila asinhronog motora konstantna i iznosi n = 300 ob/min, skicirati rukom talasne oblike napona koji se javljaju na izlazima detekcionih kompleta i naznačiti koliko iznosi njihova učestanost.

Pitanje 8:
Koliko često se resetuje dvosmerni brojač ekoderskih UP/DOWN impulsa u laboratorijskoj stanici VEKTRA pri brzini n = 300 ob/min? Obrazložiti.

Pitanje 9:
Koliko iznosi perioda odabiranja regulatora brzine u laboratorijskoj stanici VEKTRA?

Pitanje 10:
Napisati diferencne i algebarske jednačine algoritma indirektnog vektorskog upravljanja. Uzeti da je fluks nominalan i da su poznati parametri asinhronog motora.

3. Zadatak

U ovom poglavlju, naznačen je zadatak vežbe. Vežbanje se sastoji iz tri dela.

Prvi deo vežbanja čine zadaci naznačeni u Poglavlju 3.1. Vežbanje je namenjeno studentima smera EG4EV i TE5MUE. Zadaci se rešavaju putem eksperimenta na laboratorijskoj stanici VEKTRA. Za vežbanje koristiti programe: *realtime.exe*, *enc2000.exe*, *sin2000.exe*, *direct.exe* i *speed.exe*. Pomenuti programi se nalaze na PC računaru laboratorijske stanice VEKTRA u direktorijumu D:\TC\VEKTRA.

Drugi deo vežbanja čine zadaci naznačeni u Poglavlju 3.2. Vežbanje je namenjeno studentima smera EG4EV i TE5MUE. Zadaci se rešavaju putem eksperimenta na laboratorijskoj stanici VEKTRA. Za vežbanje koristi program *vektor.exe*. Pomenuti program se nalazi na PC računaru laboratorijske stanice VEKTRA u direktorijumu D:\TC\VEKTRA.

Treći deo vežbanja čine zadaci naznačeni u Poglavlju 3.3. Vežbanje je namenjeno studentima smera TE5MUE. Zadaci su programskog tipa i njihovo rešavanje ne zahteva rad na laboratorijskoj stanici VEKTRA. Zadatke treba rešiti pre dolaska na vežbu. Po dolasku na vežbu, dobijene rezultate eksperimentalno verifikovati na laboratorijskoj stanici VEKTRA. Za vežbanje je potrebno imati (i) *Simulink* model *speed.mdl* i *Matlab* komandnu datoteku *go_speed.m* (ii) Programe *demo501.exe* i *demo126.exe*. Pomenuti programi i datoteke dati su su u prilog vežbi.

3.1 Upoznavanje S/W laboratorijske stanice VEKTRA

Cilj vežbanja: Upoznavanje studenata sa S/W resursima laboratorijske stanice VEKTRA:

- ❒ *Real-time* prekidnom rutinom
- ❒ Algoritmom za čitanje inkrementalnog enkodera
- ❒ Algoritmom za generisanje obrtnih vektora
- ❒ Algoritmom indirektnog vektorskog upravljanja
- ❒ Digitalnim regulatorom brzine

U nastavku, date su smernice za izradu prvog dela vežbanja.

3.1.1 Program za generisanje *Real-Time* prekida (*realtime.exe*)

Zadatak: (i) Snimiti izlaz programa *realtime.exe* (ii) Odgovoriti na pitanja

Postupak:

- ■ Pročitati Poglavlje 5.2.1 (sadrži detaljan opis programa *realtime.exe*)
- ■ Uključiti laboratorijsku stanicu VEKTRA (vidi Poglavlje 5.3)
- ■ Pokrenuti izvršenje programa *realtime.exe*
- ■ Snimiti izlaz programa *realtime.exe* (povorku pravougaonih impulsa)
- ■ Dobijeni talasni, rukom skicirati u Izveštaju
- ■ Izaći iz programa *realtime.exe*

Pitanja:

1. Koliko iznosi učestanost prekida u programu *realtime.exe*
2. Šta je potrebno izmeniti u izvornom kodu programa *realtime.exe* da učestanost prekida bude 2 puta veća?

Odgovore na pitanja uneti u Izveštaj.

3.1.2 Program za čitanje inkrementalnog enkodera (*enc2000.exe*)

Zadatak: (i) Snimiti izlaz programa *enc2000.exe* (ii) Odgovoriti na pitanja

Postupak:

- ■ Pročitati Poglavlje 5.2.2 (sadrži detaljan opis programa *enc2000.exe*)
- ■ Uključiti laboratorijsku stanicu VEKTRA (vidi Poglavlje 5.3)
- ■ Pokrenuti izvršenje programa *enc2000.exe*
- ■ Rukom okretati vratilo asinhronog motora
- ■ Snimiti izlaz programa *enc2000.exe* (detektovanu poziciju)
- ■ Dobijeni talasni oblik, rukom skicirati u Izveštaju
- ■ Izaći iz programa *enc2000.exe*

Pitanja:

1. Naznačiti sekvencu u izvornom kodu programa *enc2000.exe* koja čita sadržaj brojača enkoderskih impulsa?
2. Objasniti ulogu kontrolnog signala ENABLE u ovoj sekvenci?

Odgovore na pitanja uneti u Izveštaj.

3.1.3 Program za generisanje obrtnih vektora (*sin2000.exe*)

Zadatak: (i) Snimiti izlaz programa *sin2000.exe* (ii) Odgovoriti na pitanja

Postupak:

- Pročitati Poglavlje 5.2.3 (detaljan opis programa *sin2000.exe*)
- Uključiti laboratorijsku stanicu VEKTRA (vidi Poglavlje 5.3)
- Pokrenuti izvršenje programa *sin2000.exe*
- Rukom okretati vratilo asinhronog motora
- Snimiti izlaz programa *sin2000.exe* (sinus detektovane pozicije)
- Dobijeni talasni oblik, rukom skicirati u Izveštaju
- Izaći iz programa *sin2000.exe*

Pitanja:

1. Naznačiti sekvencu u izvornom kodu programa *sin2000.exe* koja izračunava sinus detektovane pozicije i izračunatu vrednost ispisuje na D/A konvertor?
2. Koliko odbiraka ima sinusna tabela i kako je ona organizovana?

Odgovore na pitanja uneti u Izveštaj.

3.1.4 Program za indirektno vektorsko upravljanje asinhronim motorom sa direktnim zadavanjem momenta (*direct.exe*)

Zadatak: (i) Snimiti izlaz programa *direct.exe* (ii) Odgovoriti na pitanja

Postupak:

- Pročitati poglavlje 5.2.4 (sadrži detaljan opis programa *direct.exe*)
- Uključiti laboratorijsku stanicu VEKTRA (vidi Poglavlje 5.3)
- Pokrenuti izvršenje programa *direct.exe*
- U programu *direct.exe*, zadati odskočnu promenu struje *Iq* sa 0 na 5
- Snimiti izlaz programa *direct.exe* (brzina obrtanja vratila asinhronog motora)
- Dobijeni talasni oblik, rukom skicirati u Izveštaju
- Izaći iz programa *direct.exe*

Pitanja:

1. Kakav je karakter odskočnog odziva brzine i šta on sugeriše?
2. Naznačiti sekvencu u izvornom kodu programa *direct.exe* koja realizuje algoritam indirektnog vektorskog upravljanja asinhronim motorom?
3. Objasniti ulogu konstane SLIPGAIN u ovoj sekvenci?

Odgovore na pitanja uneti u Izveštaj.

3.1.5 Program za indirektno vektorsko upravljanje asinhronim motorom sa regulacijom brzine (*speed.exe*)

Zadatak: (i) Snimiti izlaz programa *speed.exe* (ii) Odgovoriti na pitanja

Postupak:

- Pročitati Poglavlje 5.2.5 (sadrži detaljan opis programa *speed.exe*)
- Uključiti laboratorijsku stanicu VEKTRA (vidi Poglavlje 5.3)
- Pokrenuti izvršenje programa *speed.exe*
- U programu *speed.exe*, zadati odskočnu promenu brzine od 0 do 500 [ob/min]
- Snimiti izlaz programa *speed.exe* (brzina obrtanja vratila asinhronog motora)
- Dobijeni talasni oblik, rukom skicirati u Izveštaju
- Izaći iz programa *speed.exe*

Pitanja:

1. Kakav je karakter odskočnog odziva brzine?
2. Koliko traje prelazni proces zaletanja?
3. Koliko iznosi perioda odabiranja regulatora brzine?

Odgovore na pitanja uneti u Izveštaj.

3.2 Eksperimentalni rad na laboratorijskoj stanici VEKTRA

Cilj vežbanja: Upoznavanje studenata sa izgledom talasnog oblika struje statora asinhronog motora, uobličene putem histerezisnog strujnog regulatora i uticajem temperature na tačnost rada algoritma indirektnog vektorskog upravljanja. U nastavku, date su smernice za izradu drugog dela vežbanja.

3.2.1 Snimanje valovitosti statorske struje asinhronog motora

Zadatak: (i) Snimiti talasni oblik struje statora asinhronog motora u laboratorijskoj stanici VEKTRA (ii) Odgovoriti na postavljena pitanja.

Postupak:

- Uključiti laboratorijsku stanicu VEKTRA (vidi Poglavlje 5.3)
- Pokrenuti izvršenje programa *vektor.exe* (vidi Poglavlje 6.4)
- Zadati referentnu brzinu $n = 300$ ob/min (motor je neopterećen)
- Na mernom mestu 3, snimiti struju statora asinhronog motora (vidi Poglavlje 5.1.2)
- Dobijeni talasni oblik prikazati u Izveštaju
- Ponoviti zadnje tri tačke za slučaj referentne brzine $n = 0$ ob/min.
- Izaći iz programa *vektor.exe*

Pitanja:

1. Koliko iznosi učestanost fundamentala struje statora pri brzini $n = 300$ ob/min?
2. Koliko iznosi učestanost strujnog ripla pri brzini $n = 0$ ob/min?
3. Koliko iznosi amplituda strujnog ripla pri brzini $n = 0$ ob/min?

3.2.2 Analiza uticaja promene temperature na tačnost rada algoritma indirektnog vektorskog upravljanja

Zadatak: (i) Putem eksperimenta na laboratorijskoj stanici VEKTRA, analizirati uticaj temperature na tačnost rada algoritma indirektnog vektorskog upravljanja (ii) Odgovoriti na postavljena pitanja.

Postupak:

- Uključiti laboratorijsku stanicu VEKTRA (vidi Poglavlje 5.3)
- Pokrenuti izvršenje programa *vektor.exe* (vidi Poglavlje 6.4)
- U programu *vektor.exe*, zadati referentnu brzinu n = 300 ob/min (opcija 1.)
- Kočioni asinhroni motor opteretiti strujom I_{dc} = 1.5 A (vidi Poglavlje 5.1.4)
- Na ekranu PC računara, uočiti talasni oblik pokretačkog momenta i brzine
- Ne menjajući referentnu brzinu, postepeno smanjivati konstantu klizanja SLIPGAIN (opcija 9.), od početne vrednosti SLIPGAIN = 2634 ka nuli, sa korakom 100
- Pri svakoj promeni konstante SLIPGAIN, uočiti promene na talasnom obliku pokretačkog momenta
- Konstantu SLIPGAIN smanjivati sve dok brzina naglo ne opadne (skoro na nulu)
- Kada se to desi (incidentna situacija), OBAVEZNO (u roku od 30 sekundi) izlazni napon AT smanjiti na nulu, a konstantu SLIPGAIN vratiti na početnu vrednost (SLIPGAIN = 2634)
- Struju I_{dc} smanjiti na nulu
- Po potrebi, ogled ponoviti

Pitanja:

1. Prethodni ogled simululira rad indirektno vektorski upravljanog asinhronog motora u prisustvu (a) povišene radne temperature (b) snižene radne temperature?
2. Usled čega je došlo do naglog smanjenja brzine? Obrazložiti.

3.3 Izmena S/W laboratorijske stanice VEKTRA

Cilj vežbanja: Upoznavanje studenata sa algoritmima za brzo izračunavanje trigonometrijskih funkcija SIN i COS, i uticajem koji smanjenje periode odabiranja digitalnog regulatora brzine ima na svojstva brzinske regulacione petlje. U nastavku, date su smernice za izradu trećeg dela vežbanja.

3.3.1 Izmena sinusne tabele u programu *sin2000.exe* (2000 na 501)

Zadatak: Izmeniti program *sin2000.exe* tako da sinusna tabela ima 501 odbirak ¼ sinusnog talasnog oblika. Izračunavanje trigonometrijskih funkcija SIN i COS prilagoditi novom formatu sinusne tabele. Tako dobijeni program nazvati *sin501.exe*.

Postupak:

Pre dolaska na vežbu:

- Pročitati Poglavlje 4.1.4.
- Načiniti program *sin501.exe*

- Izvorni kod programa *sin501.exe* prikazati u Izveštaju

Po dolasku na vežbu, eksperimentalno verifikovati rad programa *sin501.exe*.

U prilog vežbi, dat je program *demo501.exe* (Win32). Program izračunava SIN i COS digitalne pozicije [0 1999] na osnovu *look-up* tabele sa 501 odbirak ¼ sinusnog talasnog oblika. Studentima se preporučuje da po ugledu na *demo501.exe*, zadatak reše na PC računaru a zatim, da korisne rutine prebace u *sin501.exe*.

3.3.2 Izmena sinusne tabele u programu *sin2000.exe* (2000 na 126)

Zadatak: Izmeniti program *sin2000.exe* tako da sinusna tabela ima 126 odbiraka ¼ sinusnog talasnog oblika. Izračunavanje trigonometrijskih funkcija SIN i COS prilagoditi novom formatu sinusne tabele. Zbog redukovanog broja odbiraka u sinusnoj tabeli ($R = 2$), tačnost proračuna SIN i COS obezbediti primenom linearne interpolacije. Tako dobijeni program nazvati *sin126.exe*.

Postupak:

Pre dolaska na vežbu:

- Pročitati Poglavlje 4.1.4
- Načiniti program *sin126.exe*
- Izvorni kod programa *sin126.exe* prikazati u Izveštaju

Po dolasku na vežbu, eksperimentalno verifikovati rad programa *sin126.exe*.

U prilog vežbi, dat je program *demo126.exe* (Win32). Program izračunava SIN i COS digitalne pozicije [0 1999], na osnovu *look-up* tabele sa 126 odbiraka ¼ sinusnog talasnog oblika i linearne interpolacije. Studentima se preporučuje da po ugledu na *demo126.exe*, zadatak reše na PC računaru a zatim, da korisne rutine prebace u *sin126.exe*.

3.3.3 Izmena periode odabiranja regulatora brzine u programu *speed.exe*

Zadatak: (i) Izmeniti program *speed.exe* tako da perioda odabiranja regulatora brzine bude $T = 5$ ms. Za novu vrednost periode T, izračunati vrednosti parametra regulatora brzine *KP* i *KI* tako da propusni opseg brzinske regulacione petlje bude $f_{bw} = 3$ Hz, a odskočni odziv brzine, striktno aperiodičan (maksimalno brz, bez prebačaja). Tako dobijeni program nazvati *speed5ms.exe* (ii) Eksperimentalno verifikovati rad programa *speed5ms.exe* na laboratorijskoj stanici VEKTRA.

Postupak:

Pre dolaska na vežbu:

- Upoznati se sa *Simulink* modelom *speed.mdl* (vidi Poglavlje 6.3)
- U skladu sa zadatkom, modifikovati *speed.mdl* (dobijeni model nazvati *speed5ms.mdl*)
- Na osnovu *speed5ms.mdl*, načiniti program *speed5ms.exe*
- Izvorni kod programa *speed5ms.exe* prikazati u Izveštaju

Po dolasku na vežbu, eksperimentalno verifikovati rad programa *speed5ms.exe.*

4. Teorijski osnovi

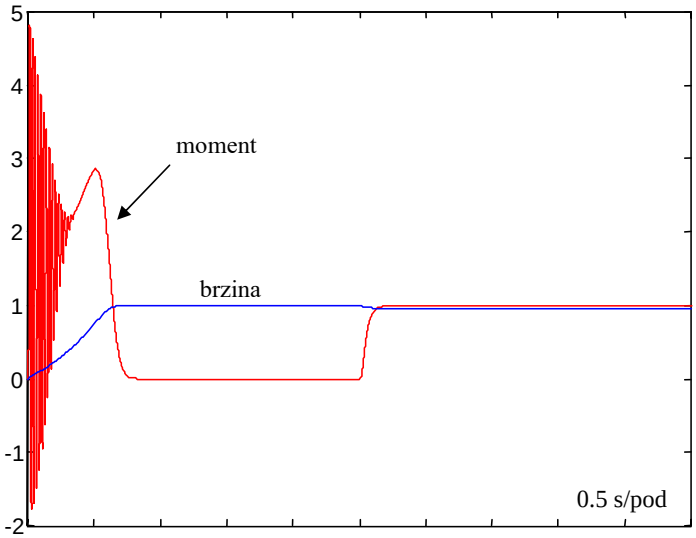
U ovom poglavlju dati su teorijski osnovi neophodni za razumevanje i uspešnu realizaciju vezbe. Izloženi su koncepti:

- ❒ Indirektnog vektorskog upravljanja asinhronim motorom
- ❒ Nelinearne strujne regulacije

4.1 Indirektno vektorsko upravljanje asinhronim motorom

Značaj asinhronih mašina u industriji je veliki, a primena raznovrsna. Po pravilu, mrežom vođena asinhrona mašina se koristi kao izvršni organ (aktuator elektromagnetnog momenta) u pogonima od kojih se zahteva pouzdan rad u uslovima teške industrijske eksploatacije i gde ne postoje strogi zahtevi u pogledu kvaliteta izlaznog momenta. Primera radi, asinhrone mašine se koriste za pogon velikih pumpnih postrojenja u vodoprivredi i naftnoj industriji. Relevantne analize načinjene u 2001 godini [1] pokazuju da asinhrone mašine troše do 50% ukupno proizvedene električne energije.

U primenama u kojima postoje strogi zahtevi u pogledu kvaliteta izlaznog momenta (servo aplikacije i električna vuča), mrežom vođena asinhrona mašina se ne koristi jer ima izrazito nepovoljne regulacione karakteristike. Ovu činjenicu najbolje ilustruje naredna simulacija. Simuliran je ogled zaletanja asinhronog motora koji se napaja mrežnim naponom, bez i u prisustvu opterećenja. Rezultat simulacije je prikazan na sl. 1.

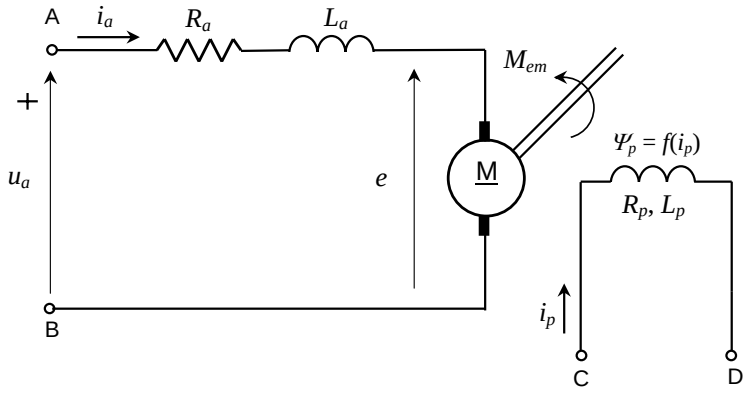


Slika 1. Odskočni odziv mrežom vođenog asinhronog motora.

Pod dejstvom mrežnog napona, motor se zaleće u praznom hodu a zatim se opterećuje nominalnim momentom. U toku zaletanja, odskočni odziv izlaznog momenta je oscilatoran kao posledica visokog reda diferencijalnih jednačina kojima se može opisati dinamički model asinhronog motora. Iz tog razloga, mrežom vođena asinhrona mašina se ne koristi u električnoj vuči i servo aplikacijama, gde se zahteva brz i precizan odziv aktuatora.

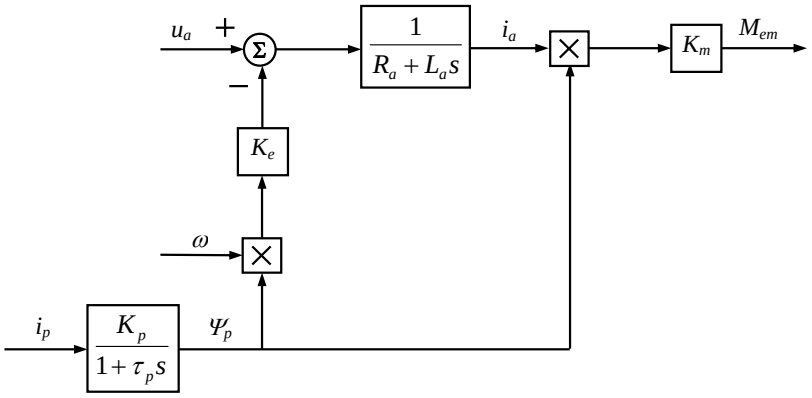
4.1.1 Svrha

Svrha indirektnog vektorskog upravljanja je da od asinhrone mašine načini linearni izvršni organ po ugledu na strujno napajanu mašinu jednosmerne struje sa nezavisnom pobudom. Podsetimo, mašina jednosmerne struje (MJS) sa nezavisnom pobudom poseduje armaturno strujno kolo i pobudno strujno kolo (vidi sl. 2).



Slika 2. Zamenska električna šema MJS sa nezavisnom pobudom.

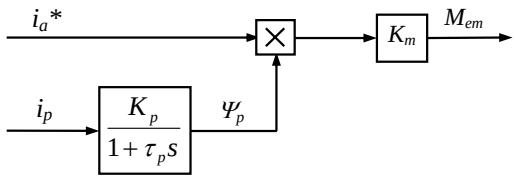
U linearnom režimu rada i pri konstanoj pobudnoj struji i_p, MJS se ponaša kao sistem prvog reda upravljan naponom u_a (vidi sl. 3).



Slika 3. Dinamički model MJS napajane iz naponskog izvora.

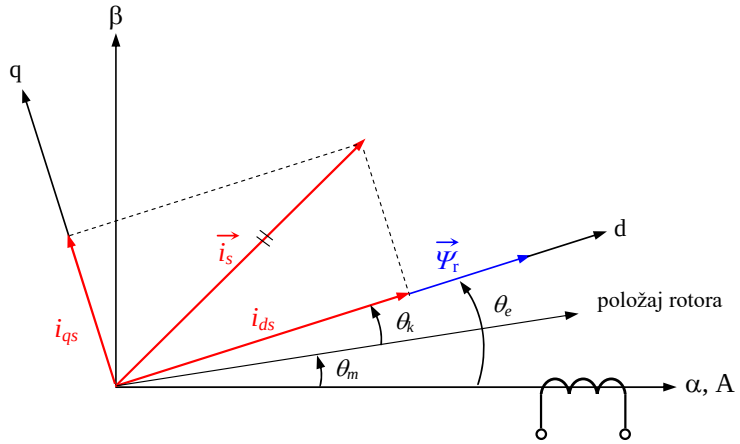
Ako se MJS napaja iz strujnog izvora, dinamički model na sl. 3 se dodatno uprošćava. Na priključne krajeve armaturnog namotaja (A i B), dovodi se jednosmerni napon u_a tako da armaturna struja i_a sledi referentni ulaz i_a*. Ako je strujni regulator dovoljno brz, opravdano je uzeti da je $i_a = i_a$* . Tada je saglasno modelu na sl. 3, izlazni moment MJS direktno proporcionalan referentnom ulazu i_a*, pa se strujno napajana MJS može smatrati linearnim izvršnim organom.

Dinamički model strujno napajane MJS sa nezavisnom pobudom prikazan je na sl. 4.



Slika 4. Dinamički model MJS napajane iz strujnog izvora.

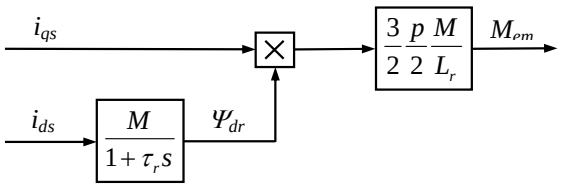
U indirektnom vektorskom upravljanju, pomoću trofaznog strujno regulisanog naponskog invertora (strujnog izvora), u asinhronu mašinu se injektuje statorska struju odgovarajuće amplitude i prostorne orijentacije (vidi sl. 5).



Slika 5. Indirektno vektorsko upravljanje asinhronim motorom.

Pomoću i_{ds} komponente statorske struje, koja se injektuje u pravcu rotorskog fluksa Ψ_r, upravlja se fluksom asinhrone mašine. Pomoću i_{qs} komponente statorske struje, koja se injektuje normalno na pravac rotoskog fluksa Ψ_r, upravlja se izlaznim momentom asinhrone mašine. Pritom, upravljanje momentom i fluksom je raspregnuto, pa je indirektno vektorski upravljana asinhroni mašina, linearni izvršni organ pogodan za primenu u električnoj vuči i servo aplikacijama.

Dinamički model indirektno vektorski upravljanog asinhronog motora prikazan je na sl. 6.



Slika 6. Dinamički model indirektno vektorski upravljanog asinhronog motora.

4.1.2 Implementacija

Jednačine naponskog balansa za stator i rotor asinhronog motora u sinhronom koordinatnom sistemu glase:

$$u_{ds} = R_s i_{ds} + \frac{\mathrm{d}\Psi_{ds}}{\mathrm{d}t} - \omega_s \Psi_{qs} \tag{1a}$$

$$u_{qs} = R_s i_{qs} + \frac{\mathrm{d}\Psi_{qs}}{\mathrm{d}t} + \omega_s \Psi_{ds} \tag{1b}$$

$$0 = R_r i_{dr} + \frac{\mathrm{d}\Psi_{dr}}{\mathrm{d}t} - \omega_k \Psi_{qr} \tag{1c}$$

$$0 = R_r i_{qr} + \frac{\mathrm{d}\Psi_{qr}}{\mathrm{d}t} + \omega_k \Psi_{dr} \tag{1d}$$

Ako se pretpostavi linearna karakteristika magnećenja, jednačine fluksnih obuhvata glase

$$\Psi_{ds} = L_s i_{ds} + M i_{dr}\,, \tag{2a}$$

$$\Psi_{qs} = L_s i_{qs} + M i_{qr}\,, \tag{2b}$$

$$\Psi_{dr} = L_r i_{dr} + M i_{ds}\,, \tag{2c}$$

$$\Psi_{qr} = L_r i_{qr} + M i_{qs}\,, \tag{2d}$$

gde su L_s i L_r, induktivnosti statora i rotora date izrazima

$$L_s = L_{\gamma s} + M \quad (M >> L_{\gamma s}), \tag{3a}$$

$$L_r = L_{\gamma r} + M \quad (M >> L_{\gamma r}). \tag{3b}$$

Zadatak indirektnog vektorskog upravljanja je da omogući što tačnije upravljanje izlaznim momentom asinhronog motora:

$$M_{em} = \vec{\Psi}_s \times \vec{i}_s = \frac{3}{2}\frac{p}{2}(\Psi_{ds} i_{qs} - \Psi_{qs} i_{ds}) = \frac{3}{2}\frac{p}{2}\frac{M}{L_r}(\Psi_{dr} i_{qs} - \Psi_{qr} i_{ds}) \tag{4}$$

Da bi se to postiglo, potrebno je ostvariti

$$\Psi_{qr} = 0. \tag{5}$$

Tada je, saglasno izrazu (4)

$$M_{em} = \frac{3}{2}\frac{p}{2}\frac{M}{L_r}\Psi_{dr} i_{qs}\,. \tag{6}$$

Klizanje pri kojem je zadovoljen uslov (5) dobija se iz izraza (1d) i (2d) i glasi

$$\omega_k = \frac{1}{\tau_r}\frac{1}{\Psi_{dr}} M i_{qs}\,, \tag{7}$$

gde je τ_r, dominanta vremenska konstanta rotora data izrazom

$$\tau_r = \frac{L_r}{R_r}\,. \tag{8}$$

Fluksom Ψ_{dr} upravlja se pomoću struje i_{ds}. Iz izraza (1c) i (2c) uz uslov (5), dobija se

$$\frac{\mathrm{d}\Psi_{dr}}{\mathrm{d}t} = -\frac{1}{\tau_r}\Psi_{dr} + \frac{1}{\tau_r}Mi_{ds} \tag{9}$$

Jednačinama (6) i (9) odgovara dinamički model prikazan na sl. 6.

Dakle, uz uslove:

- Motor je strujno napajan (propusni opseg strujne petlje je velik),
- Poznata je orijentacija rotorskog fluksa (klizanje ω_k),
- d-osa je orijentisana u pravcu rotorskog fluksa,

dinamički model asinhronog motora se svodi na model strujno napajane MJS sa nezavisnom pobudom.

Za realizaciju algoritma indirektnog vektorskog upravljanja, ključno je poznavanje prostorne orijentacije rotorskog fluksa (vidi sl. 5). Ugao θ_e se može proceniti kao

$$\theta_e = \theta_m + \theta_k\,, \tag{10}$$

gde je θ_m, ugaoni pomeraj rotora asinhrone mašine u odnosu na stator (meri pomoću davača pozicije koji se nalazi na vratilu asinhronog motora) i gde je θ_k ugao klizanja se pocenjuje na osnovu brzine klizanja (7) kao

$$\theta_k = \int_0^t \omega_k(t)\mathrm{d}t\,. \tag{11}$$

Upravljačka struktura koja realizuje algoritam indirektnog vektorskog upravljanja asinhronim motorom, prikazana je na sl. 7.

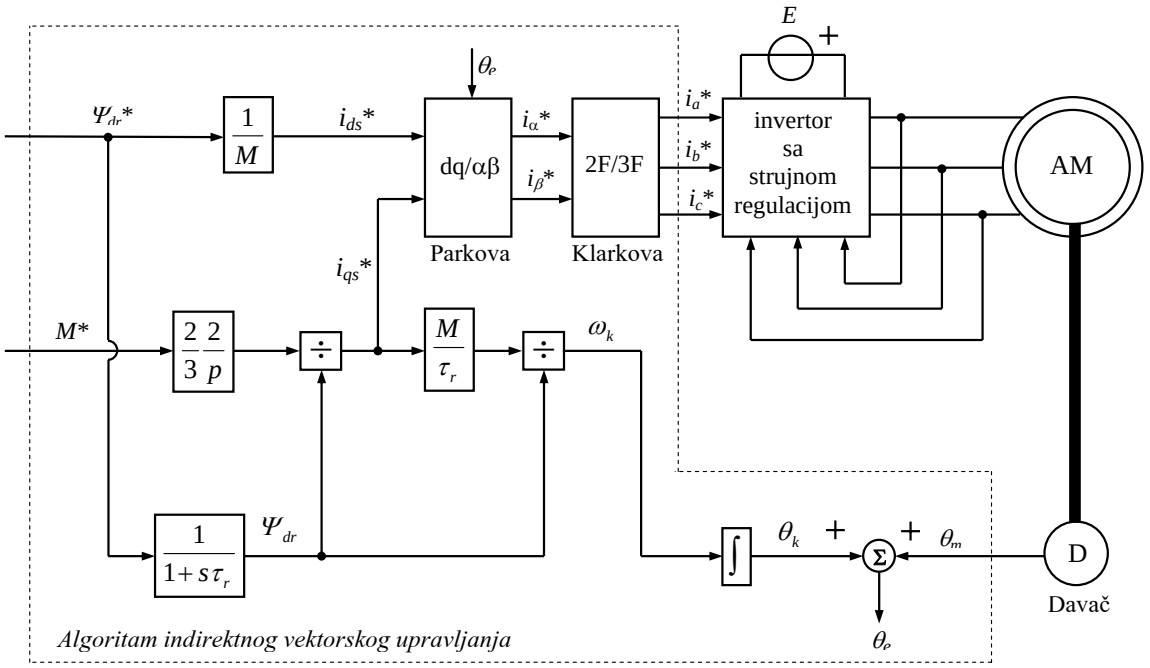
Ulazi u algoritam indirektnog vektorskog upravljanja su:

- Referentni moment M^*
- Referentni fluks Ψ_{dr}^*
- Položaj rotora asinhronog motora θ_m

Izlaz algoritma indirektnog vektorskog upravljanja su:

- Strujne reference i_a^*, i_b^* i i_c^*

Strujne reference se prosleđuju strujno regulisanom naponskom invertoru (strujnom izvoru) koji na priključnim krajevima asinhrone mašine generiše sistem trofaznih napona tako da statorske struje slede zadate strujne reference. Ako je propusni opseg strujne petlje dovoljno velik, izlazni moment asinhrone mašine jednak je referentnom momentu M^*. Tada asinhrona mašina, kao i strujno napajana MJS sa nezavisnom pobudom, postaje linearni izvršni organ pogodan za primenu u električnoj vuči i servo aplikacijama.

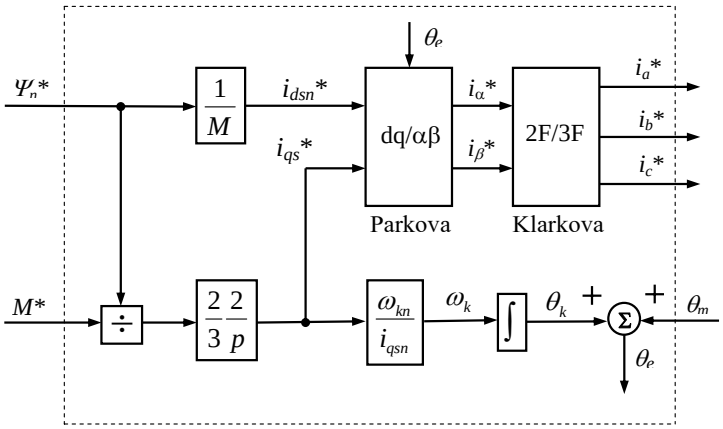


Slika 7. Blok dijagram indirektno vektorski upravljanog asinhronog motora.

U brojnim primena algoritma indirektnog vektorskog upravljanja, elektromagnetni fluks asinhrone mašine je konstantan i nominalan ($\Psi_{dr}^* = \Psi_n^*$). Tada se klizanje ω_k, saglasno izrazima (7) i (9), može proceniti kao

$$\omega_k = \omega_{kn} \frac{i_{qs}^*}{i_{qsn}}. \tag{12}$$

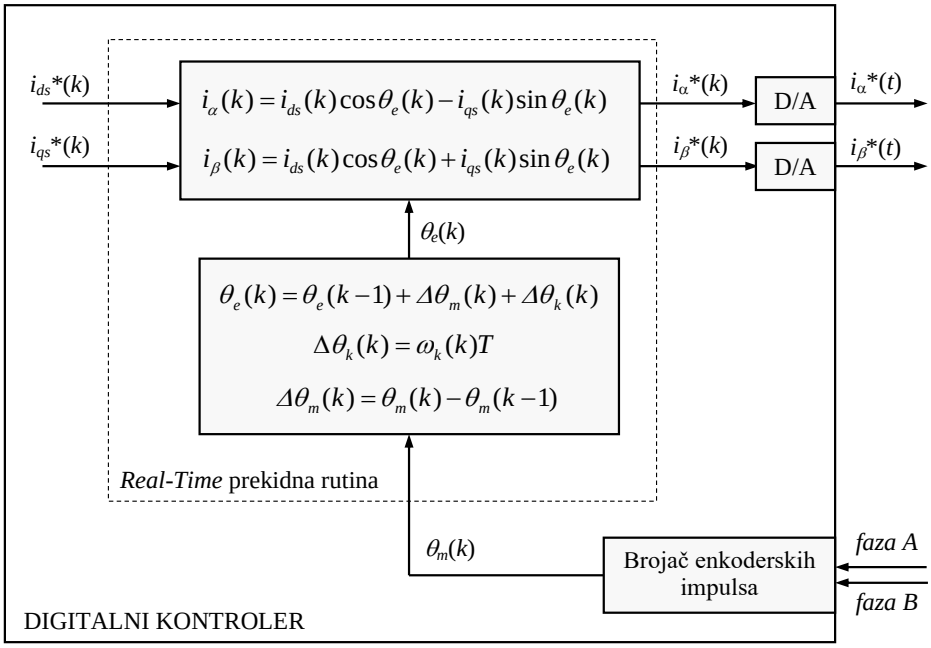
U tom slučaju, algoritam indirektnog vektorskog upravljanja ima izgled kao na sl. 8.



Slika 8. Algoritam indirektnog vektorskog upravljanja.

4.1.3 H/W zahtevi

Za realizaciju algoritma indirektnog vektorskog upravljanja prikazanog na sl. 8, potrebno je imati digitalni kontroler koji ima mogućnost generisanja *Real-Time* prekida i koji poseduje (i) brojač enkoderskih impulsa (ii) dva D/A konvertora (vidi sl. 9).



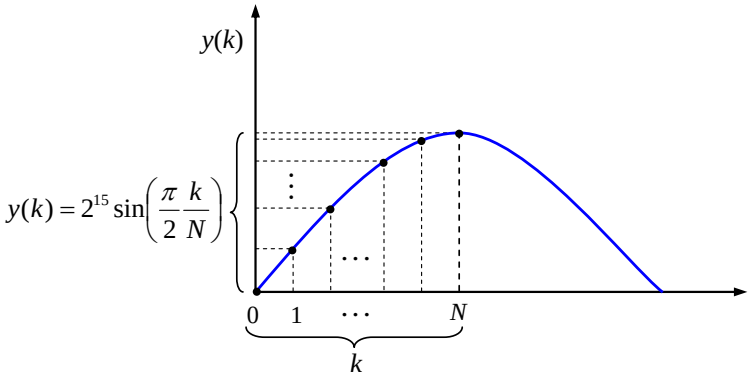
Slika 9. H/W zahtevi indirektnog vektorskog upravljanja.

Prekidna rutina u ekvidistantnim vremenskim intervalima sa periodom T, izvršava algoritam indirektnog vektorskog upravljanja. Algoritam na ulazu očitava sadržaj brojača enkoderskih impulsa $\theta_m(k)$ i digitalne strujne reference $i_{ds}^*(k)$ i $i_{qs}^*(k)$, koje stižu iz nadređene brzinske ili pozicione petlje. Na izlazu, algoritam generiše digitalne strujne reference $i_\alpha^*(k)$ i $i_\beta^*(k)$ i prosleđuje ih D/A konvertorima. Pomoću D/A konvertora, digitalne strujne reference $i_\alpha^*(k)$ i $i_\beta^*(k)$ se pretvaraju u analogne strujne reference $i_\alpha^*(t)$ i $i_\beta^*(t)$. Zadatak strujno regulisanog naponskog invertora je da zadate analogne strujne reference isprati što brže i što preciznije.

Perioda odabiranja algoritma indirektnog vektorskog upravljanja T, mora biti barem za red veličine manja od vremenske konstante rotora (8). Tada strujni regulator ima mogućnost da potisne sporopromenljivi poremećaj u vidu indukovane elektromotorne sile statora i obezbedi zahtevanu dinamiku i tačnost strujne regulacije. Tipično, vrednost periode T iznosi od x10μs do x100μs. Ovo relativno kratko vreme nameće potrebu za brzim digitalnim kontrolerima, koji u realnom vremenu mogu da obave složena izračunavanja. Pre svega, tu se misli na obrtnu transformaciju, jer podrazumeva proračun trigonometrijskih funkcija SIN i COS.

4.1.4 S/W zahtevi

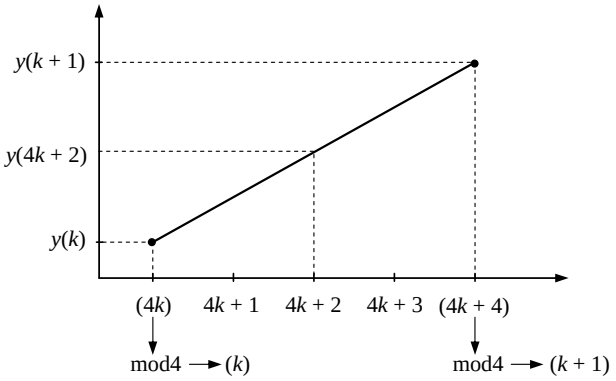
Za izračunavanje trigonometrijskih funkcija SIN i COS u algoritmu indirektnog vektorskog upravljanja, najčešće se koristi aproksimativni metod koji se zasniva na upotrebi sinusne (*look-up*) tabele. U ovom pristupu, odbirci 1/4 sinusnog talasnog oblika se u vidu tabele smeštaju u programsku memoriju digitalnog kontrolera. Način formiranja sinusne tabele prikazan je na sl. 10.



Slika 10. Tabeliranje 1/4 sinusnog talasnog oblika.

U fazi izvršenja programa, za vrednost digitalnog ekvivalenta pozicije iz opsega $[0 \quad 4N-1]$, iz sinusne tabele direktno se očitava sinus u Q15 formatu. Prednost ovog pristupa ogleda se u velikoj brzini rada. Mana je veliko zauzeće programske memorije.

Primenom linearne aproksimacije, moguće je smanjiti veličinu sinusne tabele a da se pritom, značajno ne smanji tačnost proračuna. Tipično, uzima se 2^R puta manje odbiraka ($R \geq 2$), jer tada programska realizacija linearne interpolacije zahteva deljenje faktorom 2^R, što se u digitalnom kontroleru jednostavno implementira (aritmetičko šiftovanje akumulatora za R mesta udesno). Primera radi, pri faktoru $R = 2$, za izračunavanje trigonometrijske funkcije SIN koristi se obrazac prikazan na sl. 11.



Slika 11. Izračunavanje SIN pomoću sinusne tabele i linearne aproksimacije.

4.1.4 Praktična realizacija

Polazeći od upravljačke strukture na sl. 9, u laboratorijskoj stanici VEKTRA praktično je realizovan algoritam indirektnog vektorskog upravljanja asinhronim motorom (vidi Poglavlje 6.2). Parametri asinhronog motora dati su u Poglavlju 5.1.4. Za date parametre, izračunate su dq komponente statorske struje u nominalnom radnom režimu

$$i_{dsn} = \sqrt{2} \cdot 1.37\text{A} = 1.93\text{A}\,, \tag{13a}$$

$$i_{qsn} = \sqrt{2I_{sn}^2 - I_{dsn}^2} = \sqrt{2} \cdot 1.59\text{A} = 2.24\text{A}\,. \tag{13b}$$

U poglavlju 5.1.2 je pokazano da broju 255 upisanom u registar D/A konvertora odgovara izlazni napon od +10 VDC. Otuda, procesorski ekvivalenti struja i_{dsn} i i_{qsn} uzimaju vrednosti

$$I_{DNOM} = \sqrt{2} \cdot I_{dsn} \cdot \frac{255}{5A} = 98\,, \tag{14a}$$

$$I_{QNOM} = \sqrt{2} \cdot I_{qsn} \cdot \frac{255}{5A} = 114\,. \tag{14b}$$

Polazeći od izraza (12), izvodi se procesorka konstanta SLIPGAIN, koja definiše zavisnost priraštaja ugla klizanja $\Delta\theta_k$ od struje i_{qs}*, u toku jedne periode odabiranja T

$$\text{SLIPGAIN} = \frac{\omega_{kn} T}{I_{QNOM}} K_n \frac{2\pi}{60} 2^{16} = 2108\,, \tag{15}$$

gde su:

- ω_k = 110 [ob/min], nominalna brzina klizanja
- T = 0.001 [s], perioda odabiranja
- K_n = 2000/2π, prenosni odnos inkrementalnog enkodera

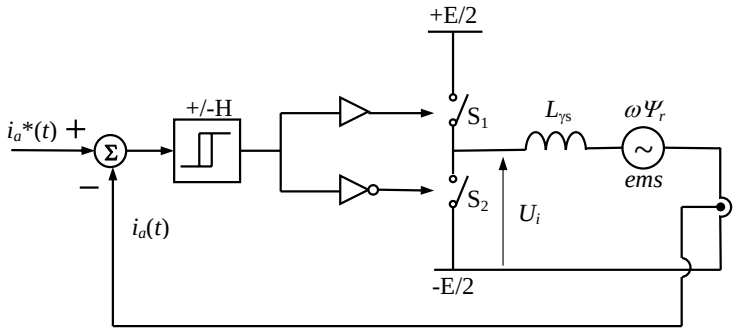
Množenjem sa 2^{16}, obezbeđuje se 32-bit tačnost proračuna.

Izvorni C kod koji realizuje algoritam indirektnog vektorskog upravljanja glasi:

```
TETA.WORD.HI += nIncPos;
TETA.WORD.HI += nIncPos;

nSlipInc = (long)SLIPGAIN*nIQref;

TETA.ALL += nSlipInc;
TETA.ALL += nSlipInc;

if(TETA.ALL > LONG1999) TETA.ALL -= LONG2000;
if(TETA.ALL < 0)     TETA.ALL += LONG2000;

nTetaE = (UINT)TETA.WORD.HI;

nSinTeta = SIN(nSineTable, nTetaE);
nCosTeta = COS(nSineTable, nTetaE);

ACC.ALL = (long)IDNOM*nCosTeta - (long)nIQref*nSinTeta;
nIalfa = ACC.WORD.HI;   // Strujne reference

ACC.ALL = (long)IDNOM*nSinTeta + (long)nIQref*nCosTeta;
nIbeta = ACC.WORD.HI;   // Strujne reference
```

4.2 Nelinearna strujna regulacija

Zahtev indirektnog vektorskog upravljanja koji se odnosi na strujno napajanje asinhronog motora može biti zadovoljen ako se za regulaciju statorske struje asinhronog motora koristi histerezisni regulator prikazan na sl. 12.



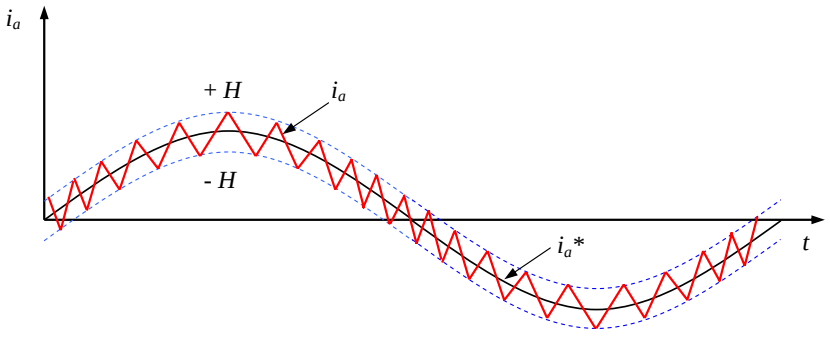
Slika 12. Nelinearna regulacija statorske struje asinhronog motora.

Princip rada:

Histerezisni regulator poredi referentnu struju i_a* i faznu struju i_a:

- Kada i_a padne na vrednost i_a* – H, pali se prekidač S_1 (struja i_a raste)
- Kada i_a dostigne vrednost i_a* + H, pali se prekidač S_2 (struja i_a opada)

Izgled statorske struje asinhronog motora, obličene putem histerezisnog strujnog regulatora, dat je na sl. 13.



Slika 13. Struja statora asinhronog motora uobličena putem histerezisnog strujnog regulatora.

Fazna struja i_a osciluje u histerezisnom prozoru oko zadate reference i_a* (vidi sl. 13) Pritom, uslov koji mora biti zadovoljen glasi

$$\frac{E}{2} > \left| ems + L_{\gamma s} \frac{\mathrm{d}i_a^*}{\mathrm{d}t} \right|_{\max} . \qquad (16)$$

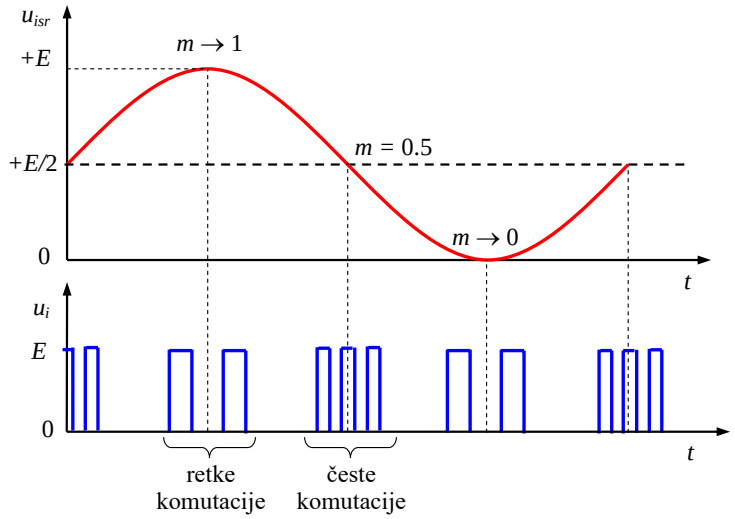
Invertor mora imati dovoljnu naponsku marginu kako ne bi došlo do izobličenja statorske struje i_a.

Amplituda strujnog ripla je konstantna i određena je širinom histerezisnog prozora H, dok se učestanost menja. U literaturi [2] je pokazano da učestanost strujnog ripla zavisi od indeksa modulacije izlaznog napona m i parametara E i $L_{\gamma s}$

$$f_{sw}(m) = \frac{E}{2L_{\gamma s}H}(m - m^2) \tag{14}$$

Prednosti:
Histerezisni regulator trenutno reaguje na strujnu grešku, pa se može smatrati da je propusni opseg strujne petlje teorijski beskonačan.

Mane:
Komutacije u invertorskom mostu nisu stacionarne. Saglasno izrazu (14), njihova učestanost zavisi od indeksa modulacije m (vidi sl. 14).



Slika 15. Nestacionarne komutacije prekidača u invertoru.

Učestanost komutacija je najmanja kada $m \to 0$ i $m \to 1$, a najveća za $m = 0.5$.

Zbog nestacionarnih komutacija, spektru izlaznog napona ne poseduje samo harmonijsku komponentu na učestanosti $1/T$ (kao kod PWM modulacije), već i mnoštvo viših harmonika. Ovo za posledicu ima povećanu buku i jako izražene elektromagnetne smetnje, koje je teško potisnuti. Komutaciona učestanost raste u oblasti malih brzina kada je indukovana *ems* statora mala. Učestanost komutacija je određena vremenom potrebnim da strujna greška napravi ekskurziju $2H$, i osim od H, zavisi i od strmine struje koja je uslovljena sa E, *ems* i $L_{\gamma s}$.

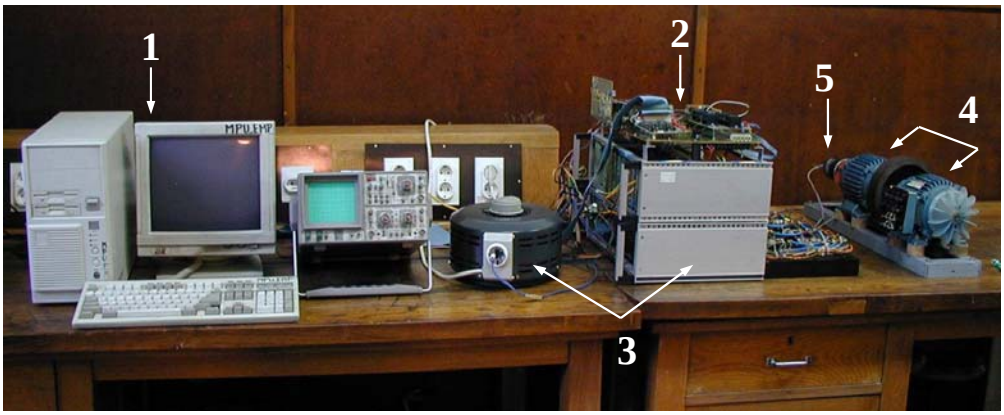
5. Praktični aspekti rada sa laboratorijskom stanicom VEKTRA

U ovom poglavlju, dat je detaljan opis laboratorijske stanice VEKTRA. Opisani su:

- ❒ H/W resursi laboratorijske stanice VEKTRA
- ❒ S/W resursi laboratorijske stanice VEKTRA
- ❒ Procedura uključenja/isključenja laboratorijske stanice VEKTRA
- ❒ Postupak merenja na laboratorijskoj stanici VEKTRA

5.1 Opis H/W resursa laboratorijske stanice VEKTRA

Izgled laboratorijske stanice VEKTRA dat je na sl. 16.



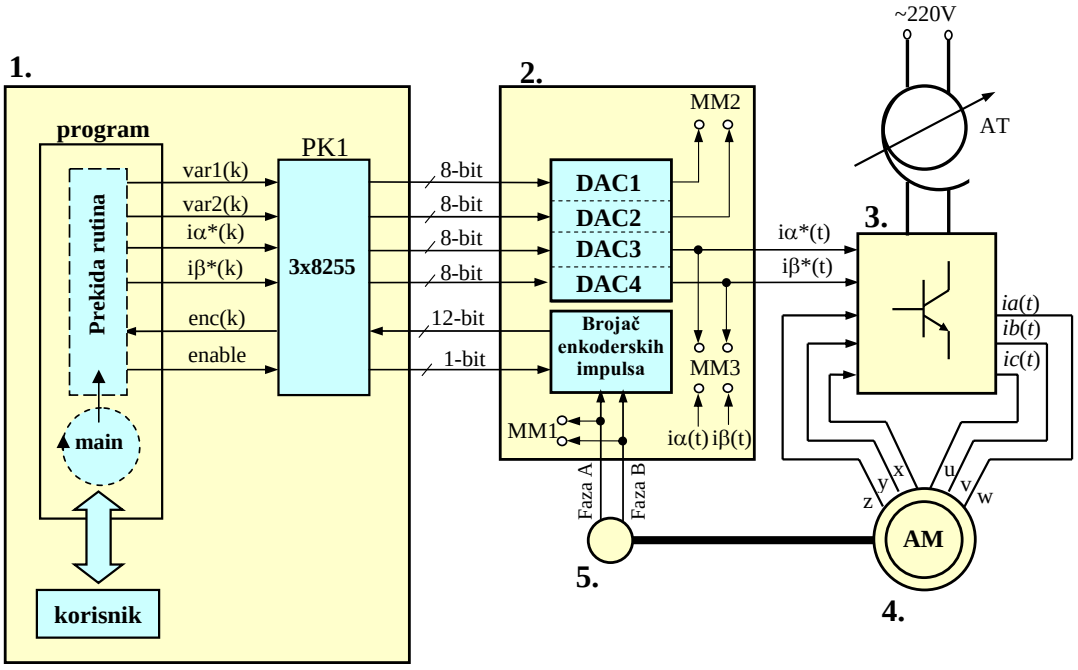
Slika 16. Laboratorijska stanica VEKTRA.

Rednim brojevima na sl. 16, označeni su H/W resursi laboratorijske stanice VEKTRA:

1. PC računar sa prilagodnom karticom 1
2. Prilagodna kartica 2
3. Strujno regulisani naponski invertor
4. Asinhroni motor i opterećenje
5. Inkrementalni optički enkoder

Osciloskop prikazan na sl. 16 deo je merne opreme laboratorijske stanice VEKTRA. Koristi se za snimanje izlaznih veličina laboratorijske stanice VEKTRA.

Funkcionalni blok dijagram laboratorijske stanice VEKTRA prikazan je na sl. 17. Ima za cilj da ukaže na uzajamnu povezanost H/W resursa laboratorijske stanice VEKTRA i osvetli tok najvažnih digitalnih i analognih signala u njoj. U tu svrhu, uzeto je da PC računar izvršava program za indirektno vektorsko upravljanje asinhronim motorom sa regulacijom brzine *speed.exe*.



Slika 17. Funkcionalni blok dijagram laboratorijske stanice VEKTRA.

Program *speed.exe* sadrži rutinu **`main`** (program niskog prioriteta) i *real-time* prekidnu rutinu **`new_timer_handler`** (vidi Poglavlje 5.2.5). Rutina **`main`** omogućava korisniku zadavanje vrednosti referentne brzine i kontrolisani izlazak iz programa. Prekidna rutina se izvršava u ekvidistantnim vremenskim intervalima sa periodom T_{SPL} = 1 ms. Ulazi u prekidnu rutinu su (i) referentna brzina (ii) digitalni ekvivalent pozicije *enc*(*k*) koji se dobija očitavanjem sadržaja brojača enkoderskih impulsa. Na osnovu ovih ulaza, u prekidnoj rutini se izvršavaju upravljačke rutine (regulacija brzine i indirektno vektorsko upravljanje) koje na izlazu daju (i) digitalne signale opšte namene *var*1(*k*) i v*ar*2(*k*) (ii) digitalne strujne reference i_α*(*k*) i i_β*(*k*) (iii) digitalni kontrolni signal *enable* koji je aktivan pri čitanju *enc*(*k*). Digitalni izlazi opšte namene se koriste za ispis digitalnih veličina na D/A konvertor. Tada se mogu posmatrati pomoću osciloskopa. U slučaju programa *speed.exe*, digitalni izlaz *var*1(*k*) je upotrebljen za ispis regulisane brzine dok se v*ar*2(*k*) ne koristi.

Prilagodna kartica 1 se nalazi na ISA magistrali PC računara. Omogućava prenos digitalnih signala *var*1(*k*), v*ar*2(*k*), i_α*(*k*), i_β*(*k*), *enc*(*k*) i *enable* između PC računara i prilagodne kartice 2. Prenos digitalnih signala se obavlja putem *flat*-kabla.

Prilagodna kartica 2 poseduje četiri D/A konvertora (DAC1 do DAC4), brojač enkoderskih impulsa i tri merna mesta (MM1 do MM3). D/A konvertori DAC1 i DAC1 vrše konverziju digitalnih signala opšte namene *var*1(*k*) i v*ar*2(*k*) u analogne signale *var*1(*t*) i v*ar*2(*t*), respektivno. Analogni izlazi *var*1(*t*) i v*ar*2(*t*) se posmatraju pomoću osciloskopa na mernom mestu 2 (MM2). D/A konvertori DAC3 i DAC3 vrše konverziju digitalnih strujnih referenci i_α*(*k*) i i_β*(*k*) u analogne strujne reference i_α*(*t*) i i_β*(*t*), respektivno. Analogne strujne reference i_α*(*t*) i i_β*(*t*) se prosleđuju strujno regulisanom naponskom invertoru. Posmatraju se pomoću osciloskopa na mernom mestu 3 (MM3).

Strujno regulisani naponski invertor generiše trofazni sistem napona na priključnim krajevima asinhronog motora tako da statorske struje $i_a(t)$, $i_b(t)$ i $i_c(t)$ odnosno, $i_\alpha(t)$ i $i_\beta(t)$ slede zadate analogne strujne reference $i_\alpha^*(t)$ i $i_\beta^*(t)$. Statorske struje $i_\alpha(t)$ i $i_\beta(t)$ posmatraju se pomoću osciloskopa na mernom mestu 3 (MM3) prilagodne kartice 2.

Inkrementalni optički enkoder sa 1000 impulsa po obrtu i dva detekciona kompleta (A i B) koristi se za merenje pozicije vratila asinhronog motora. Prilikom obrtanja vratila asinhronog motora, inkrementalni enkoder generiše impulse čija je učestanost direktno proporcionalana ugaonoj brzini obrtanja. Impulsi faza A i B inkrementalnog enkodera prosleđuju se brojaču enkoderskih impulsa. Posmatraju se pomoću osciloskopa na mernom mestu 1 (MM1) prilagodne kartice 2.

Brojač enkoderskih impulsa uobličava i broji impulse koji stižu sa faza A i B inkrementalnog optičkog enkodera. Rezultat brojanja impulsa-digitalni ekvivalent pozicije *enc*(*k*), nalazi se u dvosmernom brojaču u vidu 12-bitne digitalne reči. Prilikom čitanja digitalnog ulaza *enc*(*k*), kontrolni signal *enable* mora biti aktivan. Nakon čitanja *enc*(*k*), kontrolni signal *enable* mora biti neaktivan. Stanje signala *enable* definiše se programskim putem.

U poglavljima koja slede, dat je detaljan opis H/W resursa laboratorijske stanice VEKTRA.

5.1.1 PC računar i prilagodna kartica 1

PC računar i prilagodna kartica 1 (PK1) prikazani su na sl. 18.

Slika 18. PC računar i prilagodna kartica 1.

Električna šema veza prilagodne kartice 1 data je u Poglavlju 6.1 na sl. 28.

Prilagodna kartica 1 se nalazi na 8-bitnoj ISA magistrali PC računara. Poseduje:

- Tri paralelna porta 8255 (1P, 2P i 3P)
- Logiku za dekodovanje adresa iz opsega 0x0300 do 0x030B

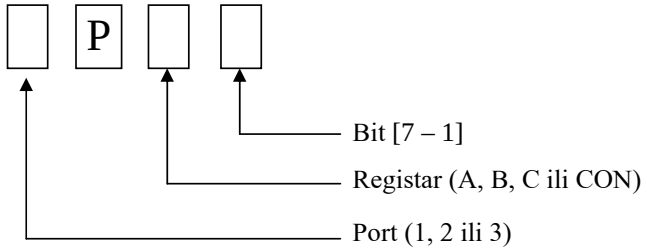
Prilagodna kartica 1 poseduje još i tajmer 8253 i dva A/D konvertora. Međutim, kako ih laboratorijskoj stanici VEKTRA ne koristi, nisu prikazani na električnoj šemi veza prilagodne kartice 1.

Konfiguracioni registri (CON) i registri podataka (A, B, C) paralelnih portova (1P, 2P i 3P) mapirani su u I/O prostor PC računara. Funkcije registara i njihove adrese naznačene su u Tabeli 1.

Tabela 1. Funkcija registara 8255 paralelnih portova i njihove adrese.

Port	Registar	Adresa	Funkcija registra (bita)
1P	A	0x0300	Ne koristi se
1P	B	0x0301	Ne koristi se
1P	C	0x0302	1PC[7 – 4] → ulaz 1PC[3 – 0] → izlaz. 1PC1 je *enable*
1P	CON	0x0303	Konfiguracioni registar
2P	A	0x0304	2PA[7 – 0] → izlaz. Šalje $i\alpha^*(k)$ na DAC3
2P	B	0x0305	2PB[7 – 0] → izlaz. Šalje $i\beta^*(k)$ na DAC4
2P	C	0x0306	2PC[7 – 0] → izlaz. Šalje *var*1(*k*) na DAC1
2P	CON	0x0307	Konfiguracioni registar
3P	A	0x0308	3PA[7 – 0] → ulaz. Čita gornja 4-bita *enc*(*k*)
3P	B	0x0309	3PB[7 – 0] → ulaz. Čita donjih 8-bita *enc*(*k*)
3P	C	0x030A	3PC[7 – 0] → izlaz. Šalje *var*2(*k*) na DAC2
3P	CON	0x030B	Konfiguracioni registar

Oznake u Tabeli 1 slede notaciju:



S obzirom da su adrese registara *n*PA, *n*PB, *n*PC i *n*PCON (*n* = 1, 2 i 3) mapirane u I/O prostor PC računara, za obraćanje ovim registarima (čitanje/upis) S/W laboratorijske stanice VEKTRA koristi makroe *inp* i *outp* programskog jezika C čiji prototipovi glase:

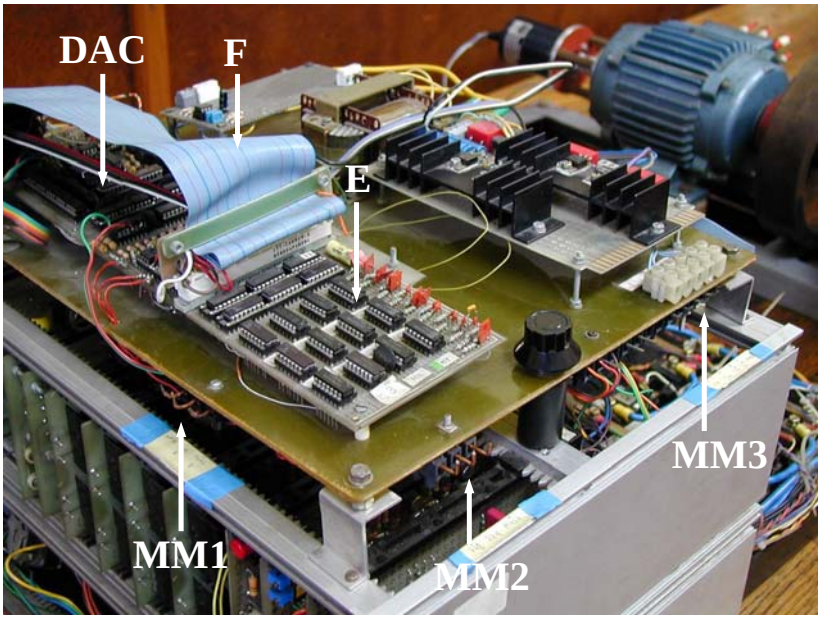
```
int  inp(int port, int byte_value);
int outp(int port, int byte_value);
```

Sekvenca programskog koda koja inicijalizuje portove prilagodne kartice 1 na način opisan u Tabeli 1 glasi:

```
outp(0x0303, 0x92);
outp(0x0307, 0x80);
outp(0x030b, 0x9a);
```

5.1.2 Prilagodna kartica 2

Izgled prilagodne kartice 2 dat je na sl. 19.



Slika 19. Izgled prilagodne kartice 2.

Električna šema veza prilagodne kartice 2 data je u Poglavlju 6.1 na sl. 29(a), 29(b) i 29(c).

Prilagodna kartica 2 poseduje:

- Četiri 8-bitna D/A konvertora (DAC1 do DAC4)
- Dvosmerni 12-bitni brojač enkoderskih impulsa
- Tri merna mesta (MM1 do MM3)
- Konektor za *flat*-kabl (F)

Na sl. 19, oblast D/A konvertora označena je sa DAC a oblast brojača enkoderskih impulsa sa E. Sa MM1, MM2 i MM3 označena su merna mesta 1, 2 i 3, respektivno.

D/A konvertori DAC1 i DAC1 vrše konverziju digitalnih izlaza opšte namene $var1(k)$ i $var2(k)$ u analogne izlaze $var1(t)$ i $var2(t)$, respektivno. Analogni izlazi $var1(t)$ i $var2(t)$ se posmatraju pomoću osciloskopa na mernom mestu 2 (MM2). Analogni izlazi D/A konvertora DAC1 i DAC2 nisu galvanski izolovani od PC računara.

D/A konvertori DAC3 i DAC3 vrše konverziju digitalnih strujnih referenci $i_\alpha^*(k)$ i $i_\beta^*(k)$ u analogne strujne reference $i_\alpha^*(t)$ i $i_\beta^*(t)$, respektivno. Analogne strujne reference $i_\alpha^*(t)$ i $i_\beta^*(t)$ se prosleđuju strujno regulisanom naponskom invertoru (vidi sl. 17). Posmatraju se pomoću osciloskopa na mernom mestu 3 (MM3). Takođe, na mernom mestu 3 mogu se posmatrati i struje statora asinhronog motora $i_\alpha(t)$ i $i_\beta(t)$. Analogni izlazi D/A konverta DAC3 i DAC4 galvanski su izolovani od PC računara pomoću optokaplera. Na ovaj način, PC računar se štiti od pojave prenapona na invertoru.

Izlazni napon 8-bitnih D/A konvertora DAC*n* uzima vrednosti iz opsega [–10 +10] VDC. Ako se u prihvatni registar D/A konvertora upiše broj 0, na njegovom izlazu će se pojaviti napon – 10 VDC. Broju 128 odgovara napon 0 VDC a broju 255, napon +10 VDC.

Brojač enkoderskih impulsa uobličava i broji impulse koji stižu sa faza A i B inkrementalnog optičkog enkodera. Enkoderski impulsi se uvode u prilagodnu karticu 2 preko optokaplera. Posmatraju se pomoću osciloskopa na mernom mestu 1 (MM1). Logika za derivaciju UP/DOWN impulsa generiše UP i DOWN impulse na osnovu uzlazne i silazne ivice jedne enkoderske faze i logičkog stanja impulsa druge enkoderske faze [2]. Uobličeni UP/DOWN impulsi se vode na ulaz dvosmernog broja. Brojač je 12-bitni i broji od 0 do 2499. Pri stanju brojača 2499, UP impuls postavlja vrednost brojača na 0; DOWN impuls dekrementira sadržaj brojača. Pri stanju brojača 0, DOWN impuls postavlja vrednost brojača na 2499; UP impuls inkrementira sadržaj brojača. Sadržaj brojača enkoderskih impulsa *enc*(*k*) postaje transparentan ako je stanje kontrolnog signala *enable* = 1 i ako je stanje brojača stabilno. Naime, interna logika brojača onemogućava aktiviranje izlaznog *latch*-a dok traje promena stanja brojača. Nakon čitanja *enc*(*k*), kontrolni signal *enable* mora biti neaktivan (*enable* = 0). Stanje kontrolnog signala *enable* definiše se programskim putem. U nastavku, naznačena je sekvenca programkog koda koja čita sadržaj brojača enkoderskih impulsa:

Comment [ZH1]: Predavanja iz MPU

```
outp(0x0302, 0x02);      /* enable = 1 */
EncHiByte = inp(0x0308); /* Cita gornjih 4 bita 12-bit brojaca */
EncLoByte = inp(0x0309); /* Cita donjih  8 bita 12-bit brojaca */
outp(0x0302, 0x00);      /* enable = 0 */
```

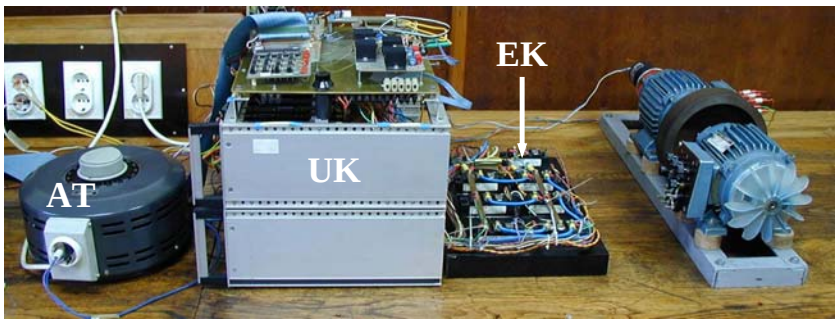
Konektor (F) uvodi *flat*-kabl u prilagodnu karticu 2. Putem *flat*-kabla, ostvarena je veza između prilagodne kartice 1 (PC računara) i prilagodne kartice 2. U poglavlju 6.1 na sl. 29(d) naznačena je funkcija pinova na konektoru (F).

Comment [ZH2]: Ovo mora sa necrta (iskopira)

5.1.3 Strujno regulisani naponski invertor

Generiše trofazni sistem napona na priključnim krajevima asinhronog motora tako da struja statora sledi zadate strujne reference $i_\alpha^*(t)$ i $i_\beta^*(t)$. Struja statora i strujne reference $i_\alpha^*(t)$ i $i_\beta^*(t)$ posmatraju se pomoću osciloskopa na mernom mestu 3 (MM3) prilagodne kartice 2.

Izgled strujno regulisanog naponskog invertora dat je na sl. 20.

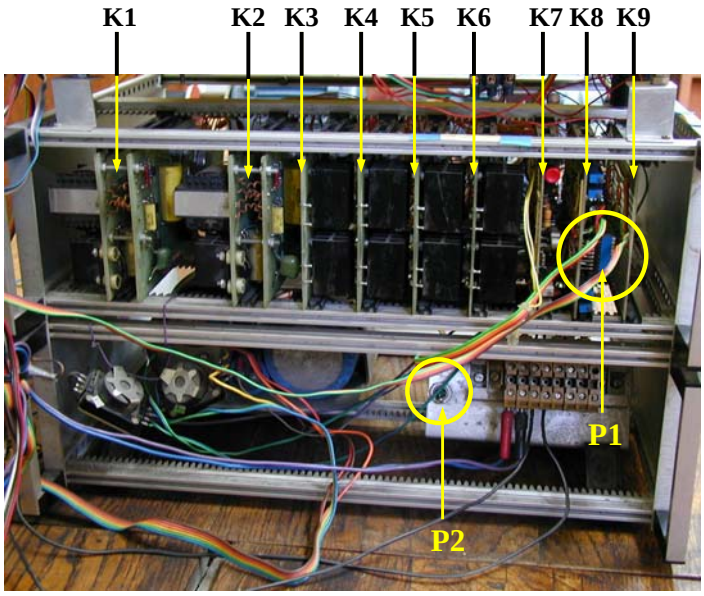


Slika 20. Strujno regulisani naponski invertor.

Strujno regulisani naponski invertor čine:

- Monofazni autotransformator (AT)
- Upravljačko kolo invertora (UK)
- Energetsko kolo invertora (EK)

Izgled upravljačkog kola invertora dat je na sl. 21.

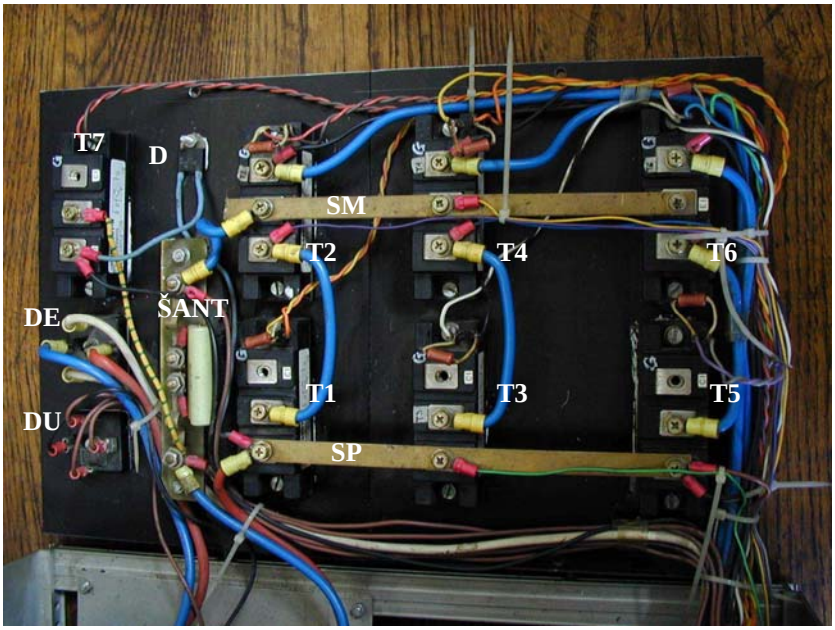


Slika 21. Upravljačko kolo invertora.

Upravljačko kolo invertora sadrži:

- Napajanje upravljačkog kola invertora (kartice K1 i K2)
- Četiri upaljačke kartice (kartice K3 do K6)
- Karticu prekostrujne i prenaponske zaštite (K7)
- Karticu strujne regulacije (K8)
- Karticu koja strujne reference $i\alpha^*(t)$ i $i\beta^*(t)$ uvodi u invertor (kartica K9)
- Prekidač za konfigurisanje moda rada invertora (P1)
- Prekidač za uključenje energetskog kola invertora (P2)

Izgled energetskog kola invertora dat je na sl. 22.



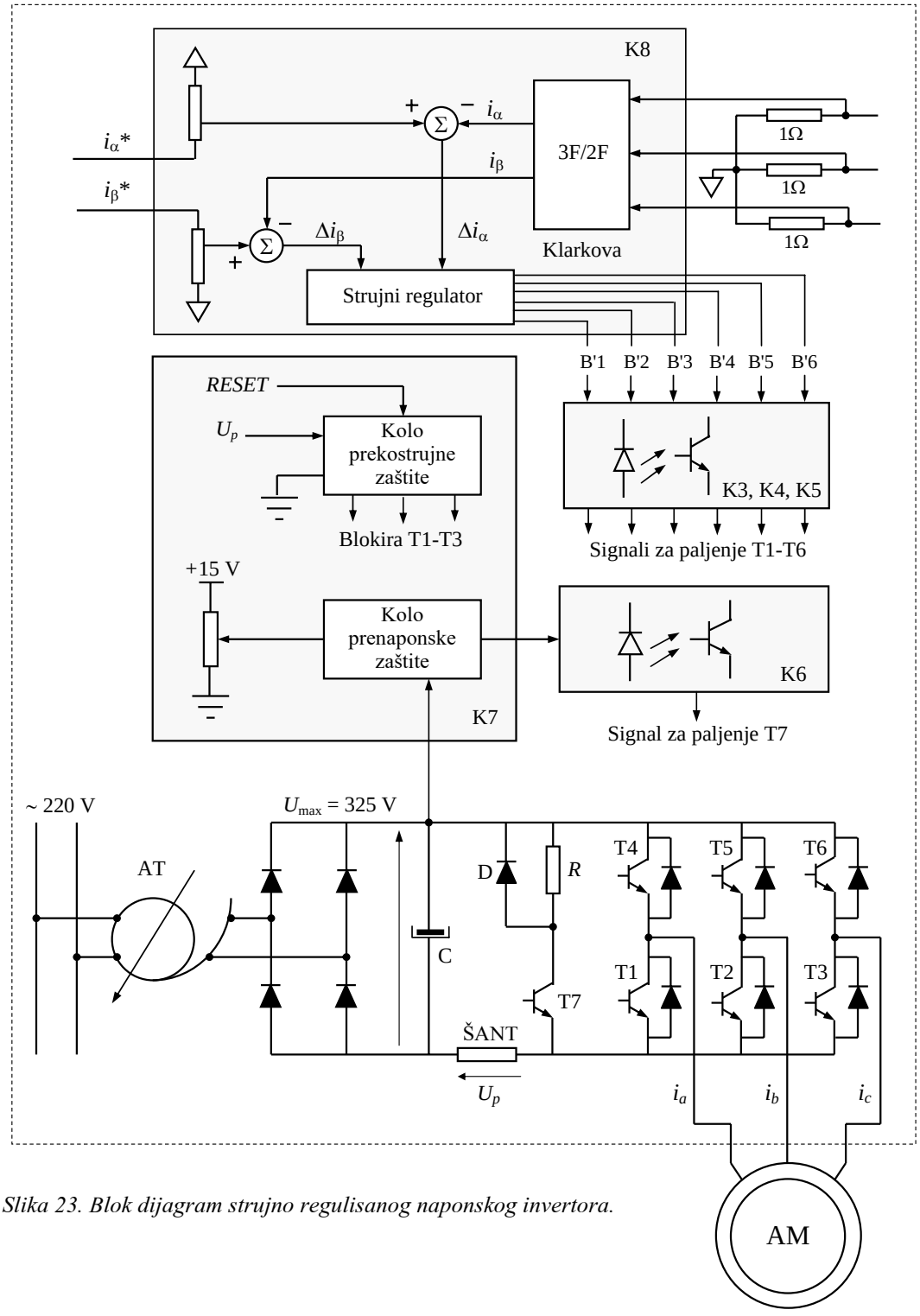
Slika 22. Energetsko kolo invertora.

Energetsko kolo invertora sadrži:

- Energetske bipolarne tranzistore invertorskog mosta (T1 do T6)
- Energetski bipolarni tranzistor za otporničko kočenje (T7)
- Diodu za otporničko kočenje (D)
- Diodni most za napajanje energetskog kola invertora (DE)
- Diodni most za napajanje upravljačkog kola invertora (DU)
- Sabirnice jednosmernog međukola (SP – plus sabirnica, SM – minus sabirnica)
- Šant otpornik za merenje struje jednosmernog međukola (ŠANT)
- Elektrolitske kondenzatore jednosmernog međukola
- Otpornik za kočenje

Elektrolitski kondenzatori i otpornik za kočenje deo su energerskog kola invertora iako se fizički nalaze u industrijskom reku, zajedno sa upravljačkim kolom invertora.

Blok šema strujno regulisanog naponskog invertora prikazana je na sl. 23.



Slika 23. Blok dijagram strujno regulisanog naponskog invertora.

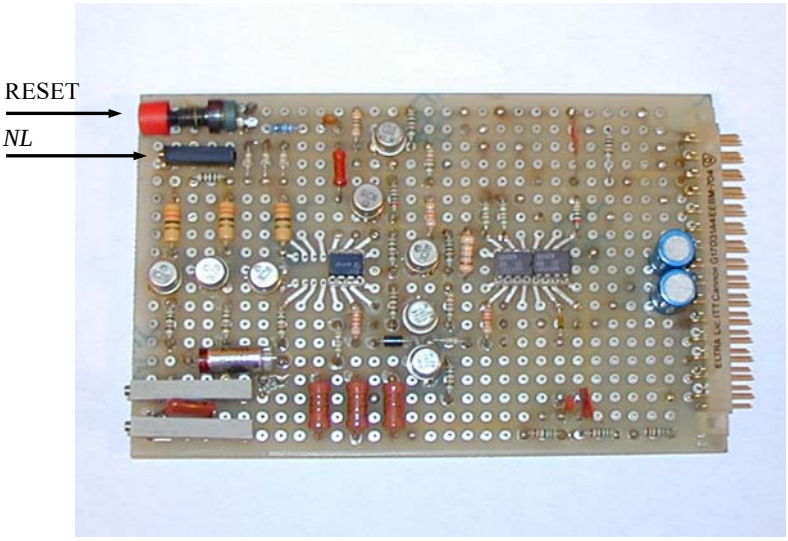
Kartice K1 i K2 su identične. Napajaju upravljačko kolo invertora. Kartice se napajaju iz zasebnog diodnog ispravljača (DU) (vidi sl. 22) sa 300VDC čime je razdvojeno napajanje upravljačkog kola i energetskog kola invertora. Na svakoj kartici se nalaze izvori jednosmernog napona, međusobno galvanski izolovani, i to po četiri izvora od +/– 5 VDC i po jedan od +/– 15 VDC. Izvori su galvanski izolovani zbog različitih naponskih nivoa na kojima se nalaze energetski tranzistori invertora. Spuštanje napona i galvanska izolacija ostvareni su pomoću DC/DC konvertora.

Električna šema veza kartica K1 i K2 data je u Poglavlju 6.1 na sl. 30.

Kartice K3 do K6 su identične. Uobličavaju, pojačavaju i optički izoluju signale za paljenje energetskih tranzistora u invertoru koji stižu sa kartice strujne regulacije (K8). Kartice K3, K4 i K5 služe za paljenje energetskih tranzistora u invertorskom mostu (T1 do T6). Kartica K6 služi za paljenje energetskog tranzistora za kočenje (T7). Svakoj kartica poseduje po dva upaljača. Na karticama K3, K4 i K5 nalaze se upaljači za tranzistore T1 do T6. Na kartici K3 se nalaze upaljači za tranzistore T1 i T2, na kartici K4 upaljači za tranzistore T3 i T4 a na kartici K5, upaljači za tranzistore T5 i T6. Na kartici K6 se nalaze dva upaljača, ali se samo jedan, donji upaljač, koristi za paljenje tranzistora za otporničko kočenje (T7). Signali za paljenje tranzistora mogu se posmatrati pomoću osciloskopa. Na prednjoj ivici svake kartice nalaze se po četiri izvoda, dva za baze i dva za emitere. Izvodi se vide na sl. 21.

Električna šema veza kartica K3 do K6 data je u Poglavlju 6.1 na sl. 31.

Kartica K7. Štiti energetsko kolo invertora od strujnog preopterećenja i ograničava vrednost napona u jednosmernom međukolu. Izgled kartice dat je na sl. 24.



Slika 24. Kartica prekostrujne i prenaponske zaštite.

Prekostrujna zaštita reaguje kada struja jednosmernog međukola invertora dostigne strujni limit. Informacija o struji jednosmernog međukola na karticu K7 stiže sa šant otpornika (ŠANT), vidi sl. 22. Napon na šantu U_P direkto je proporcionalan vrednosti struje jednosmernog međukola. Napon U_P se vodi na ulaz komparatora gde se poredi sa naponom reagovanja prekostrujne zaštite. Kada napon na šantu dostigne vrednost referentnog napona, blokiraju se donji tranzistori T1, T2 i T3, pa struja statora asinhronog motora pada na nulu (vreme opadanja statorske struje reda je nekoliko vremenskih konstanti rotorskog namotaja). Deblokada prekostrujne zaštite vrši se pritiskom na taster RESET (vidi sl. 24)

Taster RESET se nalazi na prednjoj ivici kartice K7. Crvene je boje. Vidi se i na sl. 21.

Po uključenju pogona, donji tranzistori su blokirani kao da je reagovala prekostrujna zaštita, pa je potrebno pritisnuti taster RESET da bi se pogon pokrenuo.

U slučaju da prekostrujna zaštita reaguje u toku rada, obavezno isključiti napajanje invertora i obavestiti asistenta!

Prenaponska zaštita reaguje kada napon jednosmernog međukola dostigne naponski limit definisan potenciometrom NL (vidi sl. 24). *Naponski limit je podešen na 325 VDC i ne treba ga menjati*. Kada reaguje prenaponska zaštita, uključuje se tranzistor T7 i višak energije se prazni preko otpornika za kočenje R. Kada napon jednosmernog međukola padne ispod 325 VDC, tranzistor T7 se isključuje. Otpornik R ne može da disipira veliku količinu energije, pa bi u slučaju čestih uključenja tranzistora T7, došlo do njegovog pregorevanja. Napon jednosmernog međukola tada postaje previsok, pa dolazi do proboja na elektorolitskim kondezatorima. Do proboja može doći (i) usled čestih reversa motora, kada se usled kočenja često uključuje tranzistor T7 (ii) ako je izlazni napon AT veći od 230V (napon jednosmernog međukola tada je veći od 325 VDC). Zato treba biti pažljiv prilikom puštanja u rad laboratorijske stanice VEKTRA.

Izlazni napon AT treba povećavati postepeno i NIKADA preko 230V!

Električna šema veza kartica K7 data je u Poglavlju 6.1 na sl. 32.

Ulazni signali u karticu su PSZAS_5 (napon U_P) i PNZAS_31 (napon jednosmernog međukola). Masa kartice se nalazi na potencijalu donje grane invertorskog mosta. Izlazni signali iz kartice su PC_1 do PC_4, PC_6 i PC_7. Njima se blokira rad pojačavača okidnih impulsa u karticama K3, K4 i K5 koji uključuju donje tranzistore u invertorskom mostu. Izlazni signali kola prenaponske zaštite (PC_21 i PC_22) aktiviraju rad donjeg pojačavača impulsa u kartici K6 koji upravlja tranzistorom za kočenje T7.

Kartica K8 generiše impulse za paljenje energetskih tranzistora u mostu invertora tako da komponente struje statora $i_\alpha(t)$ i $i_\beta(t)$ slede zadate strujne reference $i_\alpha^*(t)$ i $i_\beta^*(t)$.

Električna šema veza kartice K8 data je u Poglavlju 6.1 na sl. 33.

Na ulaz kartice K8 dovode se (i) referentni naponi $i_\alpha^*(t)$ i $i_\beta^*(t)$ prethodno skalirani u kartici K9 sa faktorom 2/3 (ii) naponi sa šantova koji se nalaze u zvezdištu asinhronog motora (vidi sl. 23). Zvezdište motora sa šantovima nalazi se van motora, u industrijskom reku zajedno sa upravljačkim kolom invertora.

Primenom Klarkove transformacije, na osovu $i_a(t)$, $i_b(t)$ i $i_c(t)$ dobijaju se naponi $i\alpha(t)$ i $i\beta(t)$. Podsetimo, veza između αβ-komponente statorske struje i faznih struja glasi:

$$i_\alpha = i_a - \frac{1}{2}i_b - \frac{1}{2}i_c \tag{15a}$$

$$i_\beta = \frac{\sqrt{3}}{2}(i_b - i_c) \tag{15b}$$

Histerezisni komparatori porede referentne napone $i_\alpha^*(t)$ i $i_\beta^*(t)$ i napone $i_\alpha(t)$ i $i_\beta(t)$, i zavisnosti od znaka i amplutude (strujne) greške, generišu signale za paljenje tranzistora T1 do T6. Mrtvo vreme (vreme potrebno za komutaciju dva energetska prekidača u istoj grani invertora) jednako je vremenu potrebnom da napon u tačama Z1, Z2 i Z3 promeni polaritet od -6.5VDC do +6.5VDC i obratno, i zavisi od strmine promene izlaznog napona komparatora 741 (*slew-rate* SR).

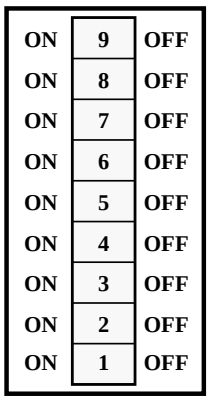
Komparator 741 ima *slew-rate* SR = 1V/μs, pa će se pri svakoj komutaciji izlaznog napona od -15VDC do +15VDC (signal za paljenje/gašenje tranzistora T1 do T6), generisati mrtvo vreme od 13μs.

Masa kartice strujne regulacije K8 nalazi se na potencijalu zvezdišta asinhronog motora. Referentni naponi $i_\alpha^*(t)$ i $i_\beta^*(t)$ se u karticu uvode preko pinova 11DAC_1 i 12_DAC2 a naponi sa šantova preko pinova koji su označeni sa SANT_A, SANT_B i SANT_C. Izlazni signali za paljenje energetskih tranzistora T1 do T7 označeni su sa PB_1 do PB_14. Preko optički izolovane veze, ovi signali se šalju do kartica K3 do K6.

Kartica K9 množi referente napone $i_\alpha^*(t)$ i $i_\beta^*(t)$ sa 2/3 i uvodi ih u karticu K8.

Faznoj struji i_a = 5A odgovara napon na šantu u_a = 5V ($R_Š$ = 1Ω) i saglasno izrazu (15), napon u_α = 3/2·5V = 7.5V. S druge strane, referentni naponi $i_\alpha^*(t)$ i $i_\beta^*(t)$ na izlazima D/A konvertra DAC3 i DAC4 pripadaju opsegu [-10V +10V]. Otuda ih je pre uvođenja u karticu strujne regulacije K8 potrebno pomnožiti faktorom 2/3. Tada referentnom naponu $i_\alpha^*(t)$ = 10V odgovara fazna struja i_a = 5A.

Prekidač P1 konfiguriše mod rada invertora. Izgled prekidača P1 dat je na sl. 25.



Slika 25. Prekidač P1 za konfigurisanje moda rada invertora.

Postoje dva moda rada: MOD1 i MOD2.

MOD1 (*default*) se koristi kada invertor koristi strujnu regulaciju odnosno, kada signale za paljenje energetskih tranzistora generiše kartica strujne regulacije K8. U ovom modu rada, kontakti prekidača P1 su stavljeni u položaj:

SW1(ON), SW2(ON), SW3(ON)	i	SW4(OFF), SW5(OFF), SW6 (OFF)

Položaj prekidača SW7, SW8 i SW9 nije bitan.

MOD2 se koristi kada invertor ne koristi strujnu regulaciju odnosno, kada signale za paljenje energetskih tranzistora generiše kartica sa 8098 mikrokontrolerom. U ovom modu rada, kontakti prekidača P1 su stavljeni u položaj:

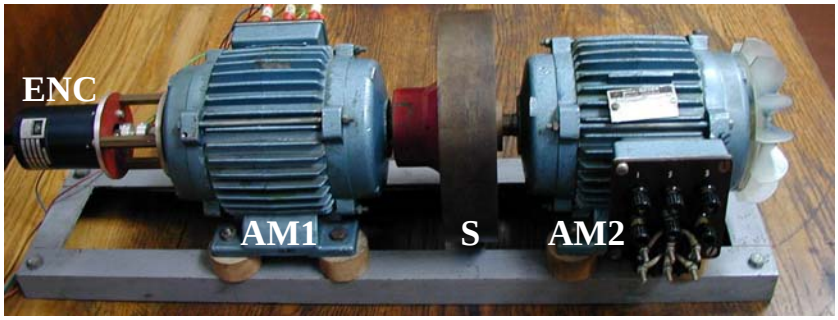
SW1(OFF), SW2(OFF), SW3(OFF)	i	SW4(ON), SW5(ON), SW6 (ON)

Položaj prekidača SW7, SW8 i SW9 nije bitan.

Prekidač P2 uključuje/isključuje energetsko kolo invertora.

5.1.4 Asinhroni motor i opterećenje

Asinhroni motor (AM1) i opterećenje (AM2) prikazani su na sl. 26.



Slika 26. Asinhroni motor i opterećenje.

Asinhroni motori AM1 i AM2 su identični. Tipa su ZK80B4 (SEVER) sa parametrima:

Sprega Y, U_n = 380V, I_n = 2.1 A, P_n = 0.75kW, cosφ = 0.72, n_{nom} = 1390 ob/min, f_n = 50Hz

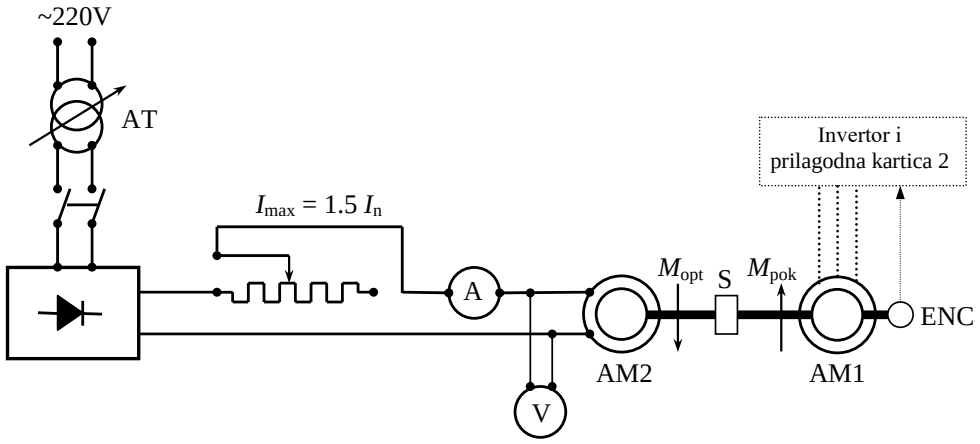
Moment inercije mehaničkog podsistema je J_{MPS} = 0.0115 kgm^2.

Parametri zamenske električne šeme asinhronog motora AM1 (AM2) su:

- Otpornost statora R_s = 8.1 Ω,
- Otpornost rotora (svedena na stranu statora) R_r = 9.6 Ω
- Induktivnost rasipanja statora $L_{\gamma s}$ = 54 mH
- Induktivnost rasipanja rotora (svedena na stranu statora) $L_{\gamma r}$ = 36.95 mH
- Induktivnost magnećenja L_m = 768.4 mH

Asinhroni motor AM1 je pokretački motor. Napaja se trofaznim naponom kojeg generiše strujno regulisani naponski invertor. Zvezdište motora sa šantovima za merenje faznih struja izvedeno je van motora i nalazi se u industrijskom reku, zajedno sa upravljačkim kolom invertora. Na vratilu AM1 montiran je davač pozicije-inkrementalni optički enkoder (ENC). Osovina AM1 je mehanički spregnuta sa osovinom AM2 pomoću mehaničke spojnice S.

Asinhroni motor AM2 je opteretni motor. Koči se jednosmernom strujom na način prikazan na sl. 27.



Slika 27. Dinamičko kočenje AM jednosmernom strujom.

Postupak kočenja AM2:

- Uključiti laboratorijsku stanicu VEKTRA
- Pokrenuti izvršenje programa *vektor.exe*
- Zadati referentnu brzinu
- Uključiti dvopolni uklopni prekidač P
- Pri konstantnoj brzini obrtanja vratila AM1, postepenim povećavanjem izlaznog napona AT, povećavati vrednost jednosmerne struje I_{DC} (ne više od I_{DC} = 1.5 I_n)
- Na monitoru PC računara, posmatrati porast I_q komponente statorske struje (momenta)
- Po završetku vežbanja, smanjiti izlazni napon AT na nulu
- Isključiti dvopolni uklopni prekidač P

Detaljan opis metoda dinamičkog kočenja AM jednosmernom strujom i uputstvo za korišćenje ogledne opreme prikazane na sl. 27, dati su u laboratorijskoj vežbi: *"Dinamičko kočenje vučnog asinhronog motora"*.

5.1.5 Inkrementalni optički enkoder

Za merenje položaja vratila asinhronog motora u laboratorijskoj stanici VEKTRA koristi se inkrementalni optički enkoder ENC (vidi sl. 26). Enkoder ima 1000 markera i dva detekciona kompleta (faze A i B). Impulsi enkoderskih faza vode se na ulaz prilagodne kartice 2, gde se poznatom metodom (i) uobličavaju u UP/DOWN impulse (ii) broje pomoću dvosmernog brojača. Detaljan opis inkrementalnog optičkog enkodera dat je u literaturi [2].

5.2 S/W laboratorijske stanice VEKTRA

S/W laboratorijske stanice VEKTRA čine:

- Program za generisanje *Real-Time* prekida (*realtime.exe*)
- Program za očitavanje inkrementalnog enkodera (*enc2000.exe*)
- Program za generisanje obrtnih vektora (*sin2000.exe*)
- Program za indirektno vektorsko upravljanje asinhronim motorom sa direktnim zadavanjem momenta (*direct.exe*)
- Program za indirektno vektorsko upravljanje asinhronim motorom sa regulacijom brzine (*speed.exe*)

Program *realtime.exe* predstavlja osnovu S/W laboratorijske stanice VEKTRA. Ilustruje kako se organizuje *Real-Time* prekidna rutina na PC računaru. Izvorni C kod programa *realtime.exe* dat je u Poglavlju 6.2 (Listing 1).

Program *enc2000.exe* predstavlja proširenje programa *realtime.exe*. Ilustruje kako se očitava inkrementalni enkoder i kako se na osnovu očitanog inkrementa pozicije formira apsolutna pozicija vratila asinhronog motora. Izvorni C kod programa *enc2000.exe* dat je u Poglavlju 6.2 (Listing 2).

Program *sin2000.exe* predstavlja proširenje programa *enc2000.exe*. Ilustruje kako se na osnovu detektovane pozicije i sinusne *look-up* tabele generišu obrtni vektori SIN i COS. Izvorni C kod programa *sin2000.exe* dat je u Poglavlju 6.2 (Listing 3).

Program *direct.exe* predstavlja proširenje programa *sin2000.exe*. Ilustruje kako se programskim putem realizuje algoritam indirektnog vektorskog upravljanja asinhronim motorom sa direktnim zadavanjem momenta. Izvorni C kod programa *direct.exe* dat je u Poglavlju 6.2 (Listing 4).

Program *speed.exe* predstavlja proširenje programa *direct.exe*. Ilustruje kako se programskim putem realizuje algoritam indirektnog vektorskog upravljanja asinhronim motorom sa regulacijom brzine. Izvorni C kod programa *speed.exe* dat je u Poglavlju 6.2 (Listing 5).

U nastavku, dat je detaljan opis S/W laboratorijske stanice VEKTRA.

5.2.1 Program za generisanje *Real-Time* prekida (*realtime.exe*)

Svrha: Prekidanje programa niskog prioriteta kako bi se u ekvidistantnim vremenskim intervalima sa periodom T_{SPL} = 1 ms, u prekidnoj rutini sinhrono izvršio skup upravljačkih rutina: (i) generisanje povorke pravougaonih impulsa (ii) ispis generisane povorke na D/A konvertor radi prikaza na osciloskopu.

Organizacija: Izvorni kod *realtime.c* (vidi Poglavlje 6.2, Listing 1) sadrži:

- Rutinu **`main`** (program niskog prioriteta)
- Prekidnu rutinu **`new_timer_handler`**

Zadatak rutine **`main`**:

- Inicijalizacija portova Prilagodne kartice 1
- Organizacija *Real-Time* prekidne rutine sa periodom T_{SPL} = 1 ms
- Organizacija izlaska iz programa

Zadatak prekidne rutine **`new_timer_handler`**:

- Generisanje povorke pravougaonih impulsa
- Ispis povorke na D/A konvertor radi prikaza na osciloskopu
- Organizacija povratka iz prekidne rutine u program niskog prioriteta

Pokretanje: Program se pokreće iz komandne linije DOS-a naredbom:

D:\TC\VEKTRA\realtime.exe <Enter>

Merenje: Izlaz programa *realtime.exe* dostupan je na mernom mestu 2 (MM2) prilagodne kartice 2 (vidi Poglavlje 5.1.2). Meri se pomoću osciloskopa na način opisan u Poglavlju 5.4.

Zaustavljanje: Program se zaustavlja naredbom:

QUIT <Enter>

5.2.2 Program za čitanje inkrementalnog enkodera (*enc2000.exe*)

Svrha: Prekidanje programa niskog prioriteta kako bi se u ekvidistantnim vremenskim intervalima sa periodom T_{SPL} = 1 ms, u prekidnoj rutini sinhrono izvršio skup upravljačkih rutina: (i) čitanje inkrementalnog enkodera (ii) ispis detektovane pozicije na D/A konvertor radi prikaza na osciloskopu.

Organizacija: Izvorni kod *enc2000.c* (vidi Poglavlje 6.2, Listing 2) sadrži:

- Rutinu **`main`** (program niskog prioriteta)
- Prekidnu rutinu **`new_timer_handler`**

Zadatak rutine **`main`**:

- Inicijalizacija portova Prilagodne kartice 1
- Organizacija *Real-Time* prekidne rutine sa periodom T_{SPL} = 1 ms
- Organizacija izlaska iz programa

Zadatak prekidne rutine **`new_timer_handler`**:

- Čitanje inkrementalnog enkodera
  - Čitanje dvosmernog brojača enkoderskih impulsa
  - Određivanje inkrementa pozicije
- Određivanje pozicije vratila asinhronog motora
- Ispis detektovane pozicije na D/A konvertor radi prikaza na osciloskopu
- Organizacija povratka iz prekidne rutine u program niskog prioriteta

Pokretanje: Program se pokreće iz komandne linije DOS-a naredbom:

D:\TC\VEKTRA\enc2000.exe <Enter>

Merenje: Izlaz programa *enc2000.exe* dostupan je na mernom mestu 2 (MM2) prilagodne kartice 2 (vidi Poglavlje 5.1.2). Meri se pomoću osciloskopa na način opisan u Poglavlju 5.4.

Zaustavljanje: Program se zaustavlja naredbom:

QUIT <Enter>

5.2.3 Program za generisanje obrtnih vektora (*sin2000.exe*)

Svrha: Prekidanje programa niskog prioriteta kako bi se u ekvidistantnim vremenskim intervalima sa periodom T_{SPL} = 1 ms, u prekidnoj rutini sinhrono izvršio skup upravljačkih rutina: (i) čitanje inkrementalnog enkodera (ii) generisanje obrtnih vektora (iii) ispis obrtnih vektora na D/A konvertor radi prikaza na osciloskopu.

Organizacija: Izvorni kod *sin2000.c* (vidi Poglavlje 6.2, Listing 3) sadrži:

- Rutinu **`main`** (program niskog prioriteta)
- Prekidnu rutinu **`new_timer_handler`**

Zadatak rutine **`main`**:

- Generisanje sinusne tabele
- Inicijalizacija portova Prilagodne kartice 1
- Organizacija *Real-Time* prekidne rutine sa periodom T_{SPL} = 1 ms
- Organizacija izlaska iz programa

Zadatak prekidne rutine **`new_timer_handler`**:

- Čitanje inkrementalnog enkodera
  - Čitanje dvosmernog brojača enkoderskih impulsa
  - Određivanje inkrementa pozicije
- Određivanje pozicije vratila asinhronog motora
- Generisanje obrtnih vektora
- Ispis obrtnih vektora na D/A konvertor radi prikaza na osciloskopu
- Organizacija povratka iz prekidne rutine u program niskog prioriteta

Pokretanje: Program se pokreće iz komandne linije DOS-a naredbom:

D:\TC\VEKTRA\sin2000.exe <Enter>

Merenje: Izlaz programa *sin2000.exe* dostupan je na mernom mestu 2 (MM2) prilagodne kartice 2 (vidi Poglavlje 5.1.2). Meri se pomoću osciloskopa na način opisan u Poglavlju 5.4.

Zaustavljanje: Program se zaustavlja naredbom:

QUIT <Enter>

5.2.4 Program za indirektno vektorsko upravljanje asinhronim motorom sa direktnim zadavanjem momenta (*direct.exe*)

Svrha: Prekidanje programa niskog prioriteta kako bi se u ekvidistantnim vremenskim intervalima sa periodom T_{SPL} = 1 ms, u prekidnoj rutini sinhrono izvršio skup upravljačkih rutina: (i) čitanje inkrementalnog enkodera (ii) određivanje brzine obrtanja vratila asinhronog motora (iii) indirektno vektorsko upravljanje asinhronim motorom (iv) ispis brzine na D/A konvertor radi prikaza na osciloskopu.

Organizacija: Izvorni kod *direct.c* (vidi Poglavlje 6.2, Listing 4) sadrži:

- Rutinu **`main`** (program niskog prioriteta)
- Prekidnu rutinu **`new_timer_handler`**

Zadatak rutine **`main`**:

- Generisanje sinusne tabele
- Inicijalizacija portova Prilagodne kartice 1
- Organizacija *Real-Time* prekidne rutine sa periodom T_{SPL} = 1 ms
- Organizacija izlaska iz programa i unos referentne *Iq*-komponene statorske struje

Zadatak prekidne rutine **`new_timer_handler`**:

- Čitanje inkrementalnog enkodera
  - Čitanje dvosmernog brojača enkoderskih impulsa
  - Određivanje inkrementa pozicije
- Određivanje brzine obrtanja vratila asinhronog motora
- Indirektno vektorsko upravljanje asinhronim motorom
- Ispis brzine na D/A konvertor radi prikaza na osciloskopu
- Organizacija povratka iz prekidne rutine u program niskog prioriteta

Pokretanje: Program se pokreće iz komandne linije DOS-a naredbom:

D:\TC\VEKTRA\direct.exe <Enter>

Merenje: Izlaz programa *direct.exe* je dostupan na mernom mestu 2 (MM2) prilagodne kartice 2 (vidi Poglavlje 5.1.2). Meri se pomoću osciloskopa na način opisan u Poglavlju 5.4.

Zaustavljanje: Program se zaustavlja naredbom:

QUIT <Enter>

5.2.5 Program za indirektno vektorsko upravljanje asinhronim motorom sa regulacijom brzine (*speed.exe*)

Svrha: Prekidanje programa niskog prioriteta kako bi se u ekvidistantnim vremenskim intervalima sa periodom T_{SPL} = 1 ms, u prekidnoj rutini sinhrono izvršio skup upravljačkih rutina: (i) čitanje inkrementalnog enkodera (ii) određivanje brzine obrtanja vratila asinhronog motora (iii) digitalna regulacija brzine (iv) indirektno vektorsko upravljanje asinhronim motorom (v) ispis digitalnih strujnih referenci i regulisane brzine na D/A konvertore.

Organizacija: Izvorni kod *speed.c* (vidi Poglavlje 6.2, Listing 5) sadrži:

- Rutinu **`main`** (program niskog prioriteta)
- Prekidnu rutinu **`new_timer_handler`**

Zadatak rutine **`main`**:

- Generisanje sinusne tabele
- Inicijalizacija portova Prilagodne kartice 1
- Organizacija *Real-Time* prekidne rutine sa periodom T_{SPL} = 1 ms
- Organizacija izlaska iz programa i unos referentne brzine

Zadatak prekidne rutine **`new_timer_handler`**:

- Čitanje inkrementalnog enkodera
  - Čitanje dvosmernog brojača enkoderskih impulsa
  - Određivanje inkrementa pozicije
- Određivanje brzine obrtanja vratila asinhronog motora

- Digitalna regulacija brzine
- Indirektno vektorsko upravljanje asinhronim motorom
- Ispis strujnih referenci na D/A konvertore DAC3 i DAC4, respektivno
- Ispis regulisane brzine na D/A konvertor DAC1 radi prikaza na osciloskopu

Pokretanje: Program se pokreće iz komandne linije DOS-a naredbom:

D:\TC\VEKTRA\speed.exe <Enter>

Merenje: Izlaz programa *speed.exe* dostupan je na mernom mestu 2 (MM2) prilagodne kartice 2 (vidi Poglavlje 5.1.2). Meri se pomoću osciloskopa na način opisan u Poglavlju 5.4.

Zaustavljanje: Program se zaustavlja naredbom:

QUIT <Enter>

5.3 Procedura uključenja/isključenja laboratorijske stanice VEKTRA

Procedura uključenja:

- Uključiti glavni prekidač na razvodnoj tabli u sobi 40 (nosi oznaku LAB 40a)
- Uključiti prekidače na radnom stolu laboratorijske stanice VEKTRA
- Uključiti PC računar
- Pokrenuti izvršenje tekućeg programa
- Obrtni kontakt AT staviti u položaj 30V
- Resetovati prekostrujnu zaštitu kartice strujne regulacije (vidi Poglavlje 5.1.3)
- Obrtni kontakt AT staviti u položaj 220V

Laboratorijska stanica je sada spremna za rad!

Od opisane procedure odstupiti samo u slučaju kada tekući program ne zahteva rad invertora, što je slučaj sa programima: *realtime.exe, enc2000.exe* i *sin2000.exe*.

U ovim slučajevima, preskočiti zadnje tri tačke u proceduri uključenja!

Procedura isključenja:

- Obrtni kontakt AT staviti u položaj 0V
- Obustaviti izvršenje tekućeg programa
- Isključiti PC računar
- Isključiti prekidače na radnom stolu laboratorijske stanice VEKTRA
- Isključiti glavni prekidač na razvodnoj tabli u sobi 40 (nosi oznaku Lab 40a)

Laboratorijska stanica VEKTRA je sada isključena!

5.4 Postupak merenja na laboratorijskoj stanici VEKTRA

Na prilagodnoj kartici 2 (vidi sl. 19), nalaze se tri merna mesta:

- Merno mesto 1 (MM1)
- Merno mesto 2 (MM2)

- Merno mesto 3 (MM3)

Na MM1 posmatraju se impulsi faza A i B inkrementalnog enkodera.

Na MM2 posmatraju se digitalni izlazi opšte namene var1(t) i var2(t) (izlazi D/A konvertora DAC1 i DAC2).

Na MM3 posmatraju se referentne struje $i_\alpha^*(t)$ i $i_\beta^*(t)$ (izlazi D/A konvertora DAC3 i DAC4) i stvarne struje $i_\alpha(t)$ i $i_\beta(t)$.

Za merenje se koristi digitalni osciloskop prikazan na sl. 16. Sonde digitalnog osciloskopa se priključuju na izvode MM1, MM2 i MM3. Ispod svakog mernog mesta, nalazi se nalepnica sa odgovarajućom oznakom:

Na MM1: faza A, faza B, MASA.

Na MM2: 774, 778, MASA

Na MM3: i_α^*, i_β^*, i_α, i_β, MASA

Pri merenju, osciloskop staviti u digitalni (STORAGE) mod. Tada se promena posmatranog talasnog oblika može zaustaviti pritiskom na dugme HOLD.

6. Prilozi

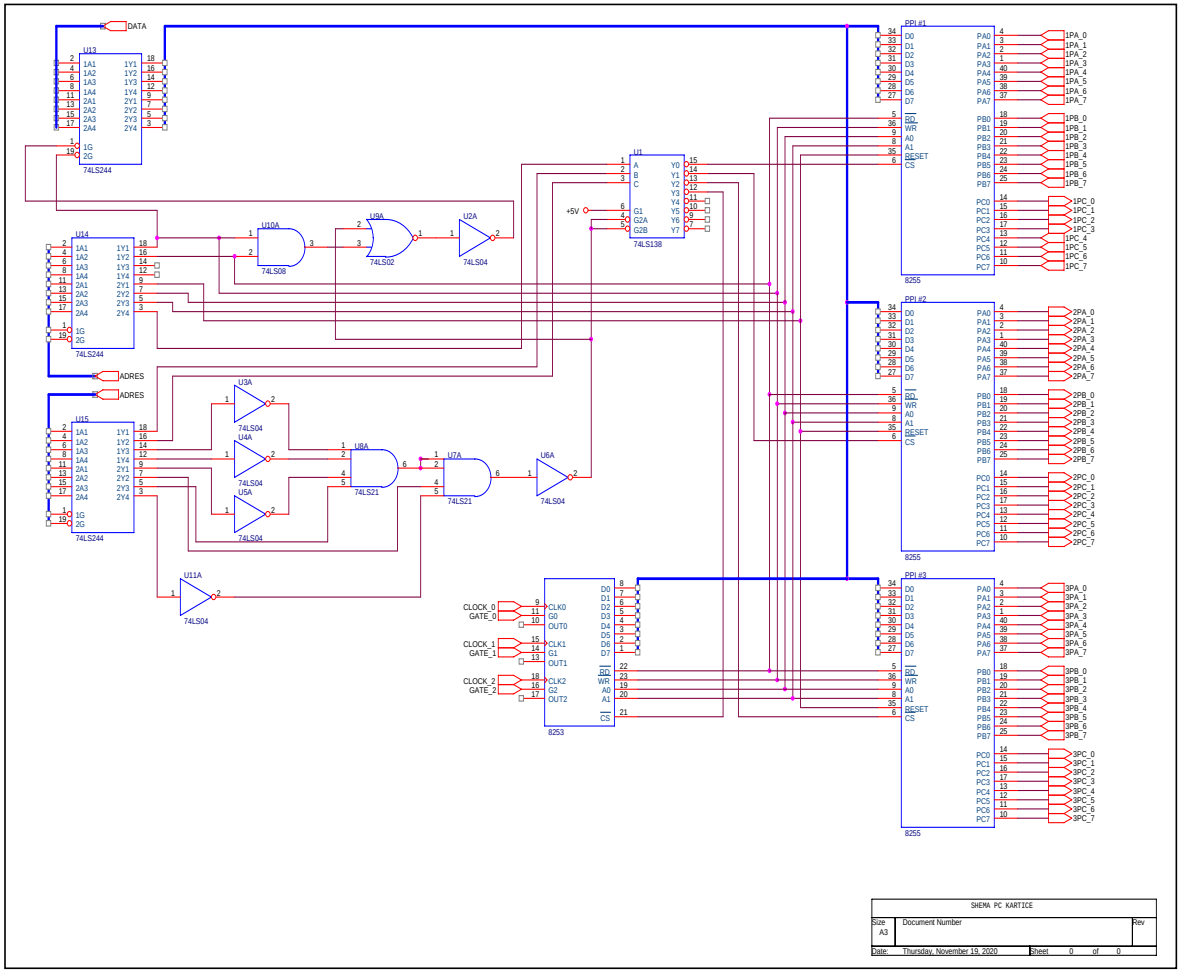
U prilog vežbi dati su:

- Električne šeme veza H/W laboratorijske stanice VEKTRA
- Izvorni kod S/W laboratorijske stanice VEKTRA
- *Simulink* model brzinske regulacione petlje *speed.mdl*
- Opis programa *vektor.exe*

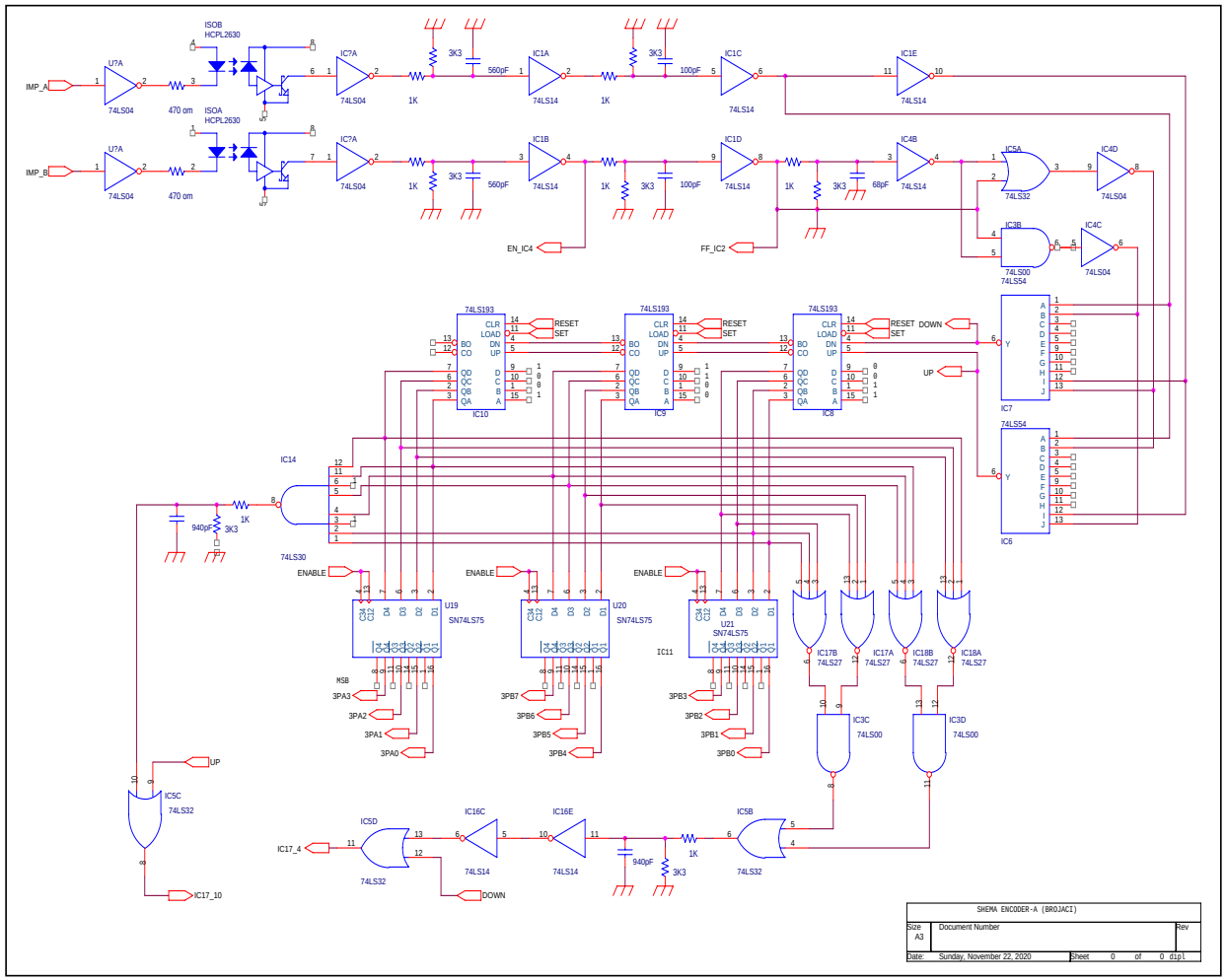
6.1 Električne šeme veza H/W laboratorijske stanice VEKTRA

- Slika 28: Električna šema veza prilagodne kartice 1
- Električna šema veza prilagodne kartice 2
  - Slika 29(a): Logika za derivaciju i brojanje enkoderskih impulsa
  - Slika 29(b): Logika za čitanje dvosmernog brojača enkoderskih impulsa
  - Slika 29(c): Oblast D/A konvertora
  - Slika 29(d): Konektor za *flat*-kabl
- Slika 30: Električna šema veza kartica K1 i K2 (Napajanje upravljačkog kola invertora)
- Slika 31: Električna šema veza upaljačkih kartica K3 do K6
- Slika 32: Električna šema veza kartice K7 (Kartica prekostrujne i prenaponske zaštita)
- Slika 33: Električna šema veza kartice K8 (Kartica strujne regulacije)
- Slika 34: Električna šema veza invertorskog mosta

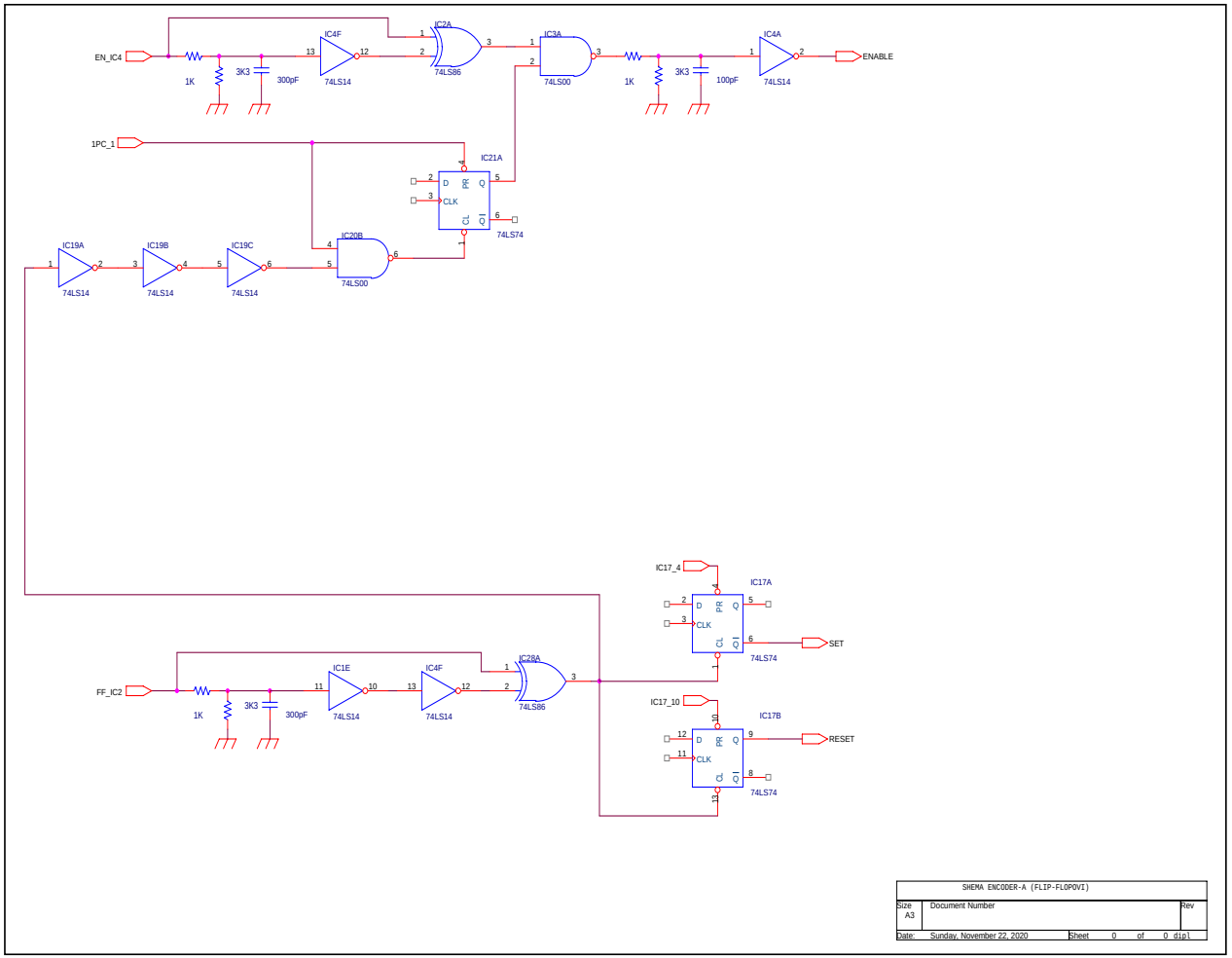
Slika 28. Električna šema veza prilagodne kartice 1.



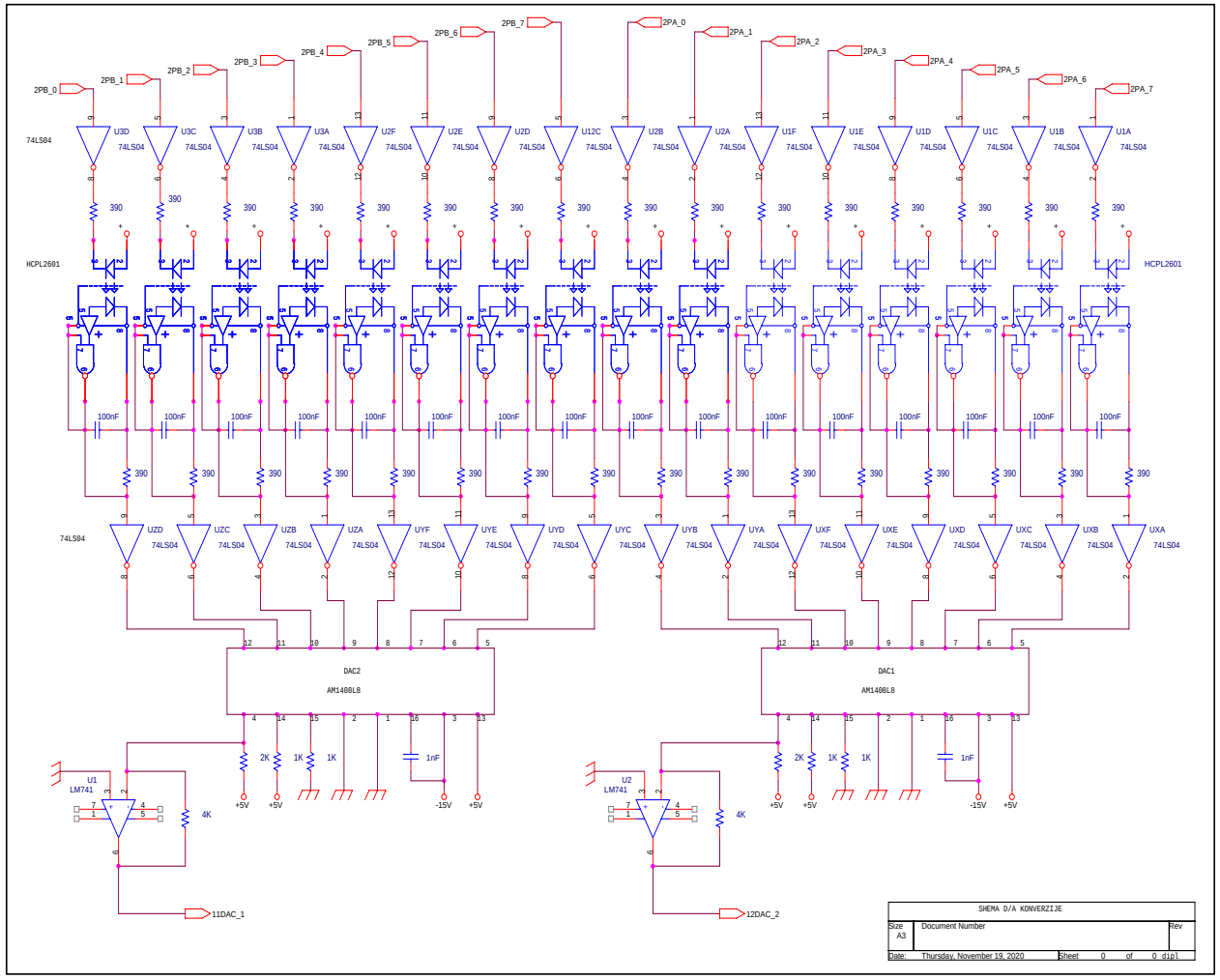
Slika 29(a). Električna šema veza prilagodne kartice 2 (Logika za derivaciju i brojanje enkoderskih impulsa).



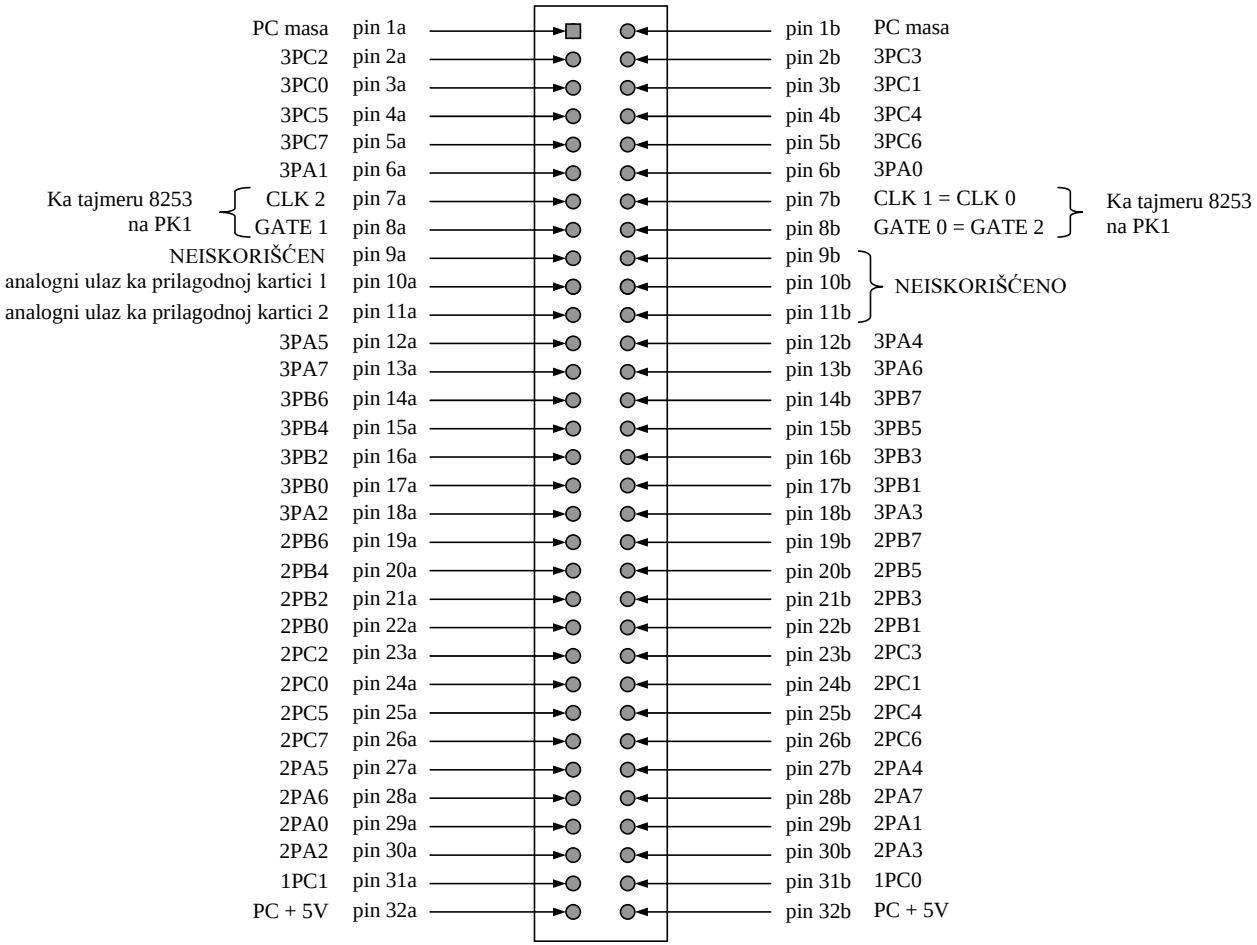
Slika 29(b). Električna šema veza prilagodne kartice 2 (Logika za čitanje dvosmernog brojača enkoderskih impulsa).



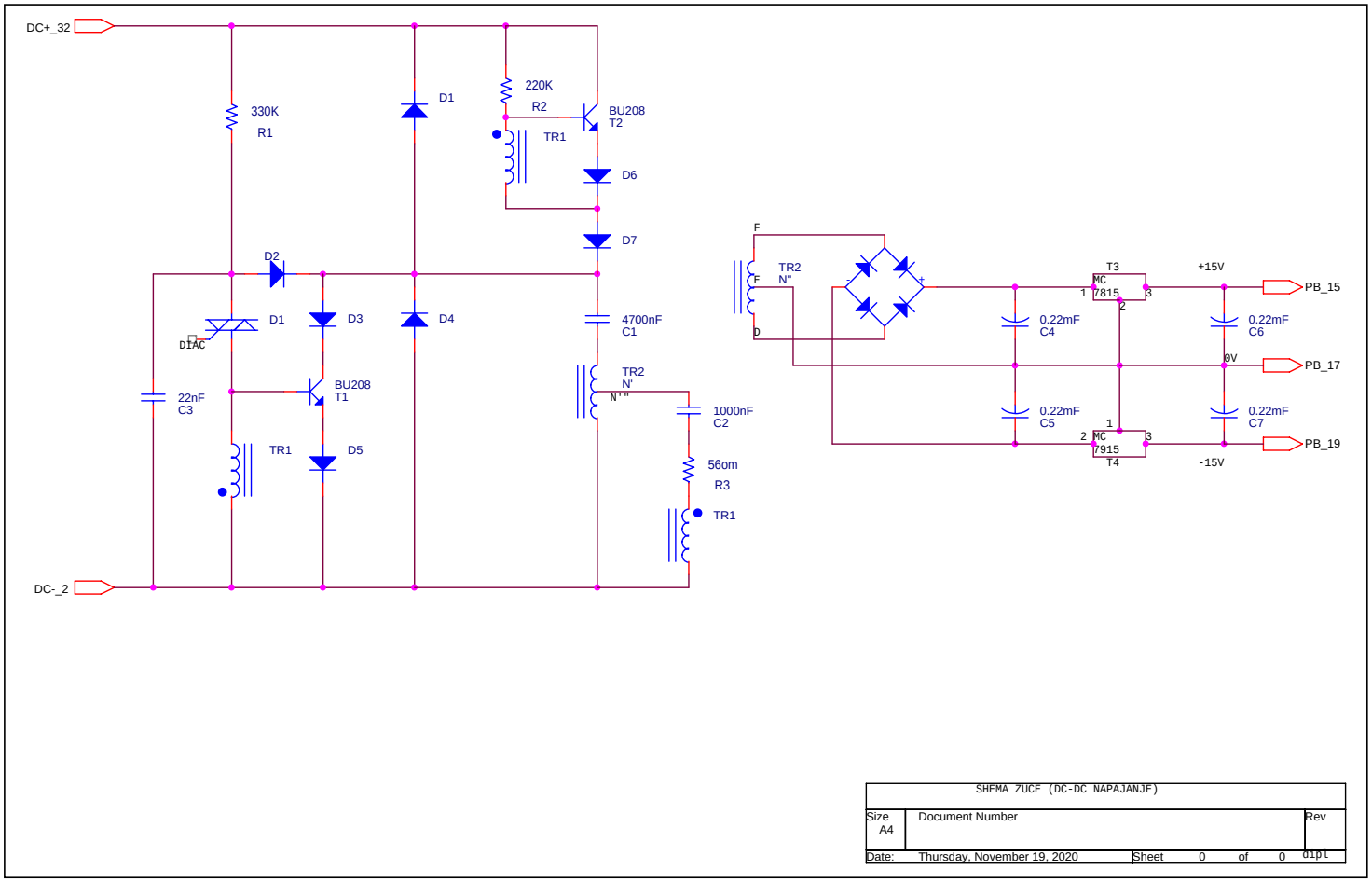
Slika 29(c). Električna šema veza prilagodne kartice 2 (Oblast D/A konvertora).



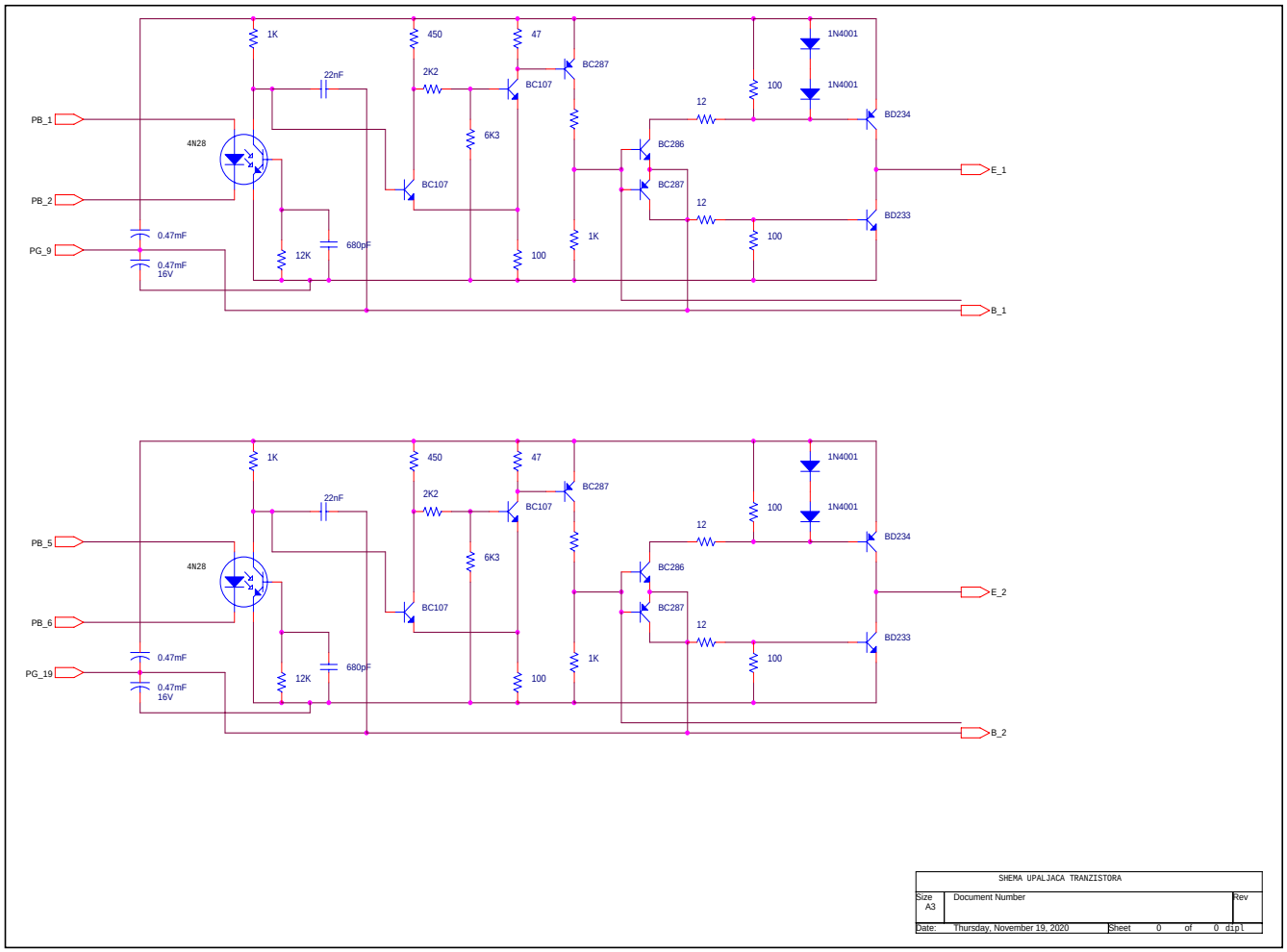
Slika 29(d). Električna šema veza prilagodne kartice 2 (Konektor za *flat*-kabl).



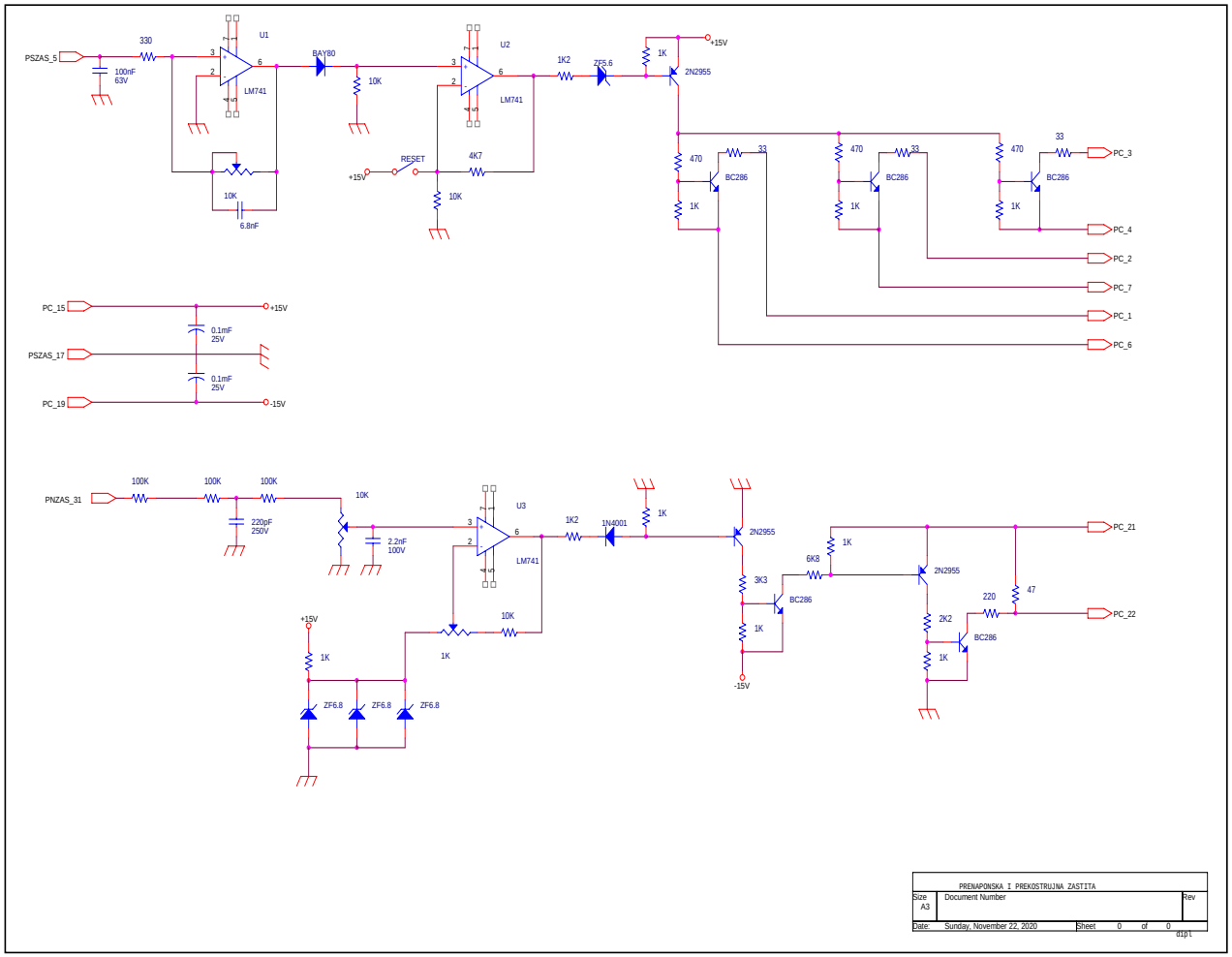
Slika 30. Električna šema veza kartica K1 i K2 (Napajanje upravljačkog kola invertora).



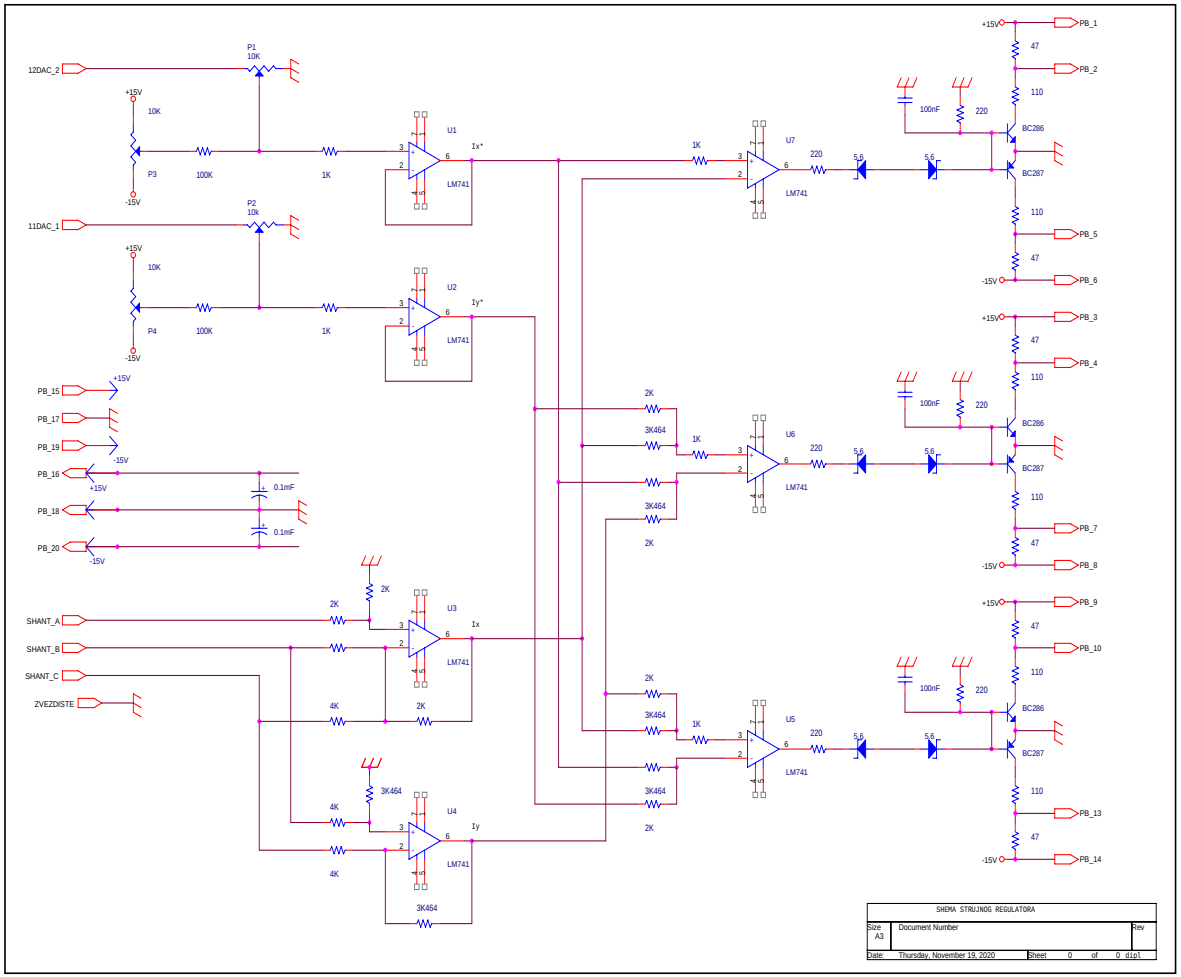
Slika 31. Električna šema veza upaljačkih kartica K3 do K6



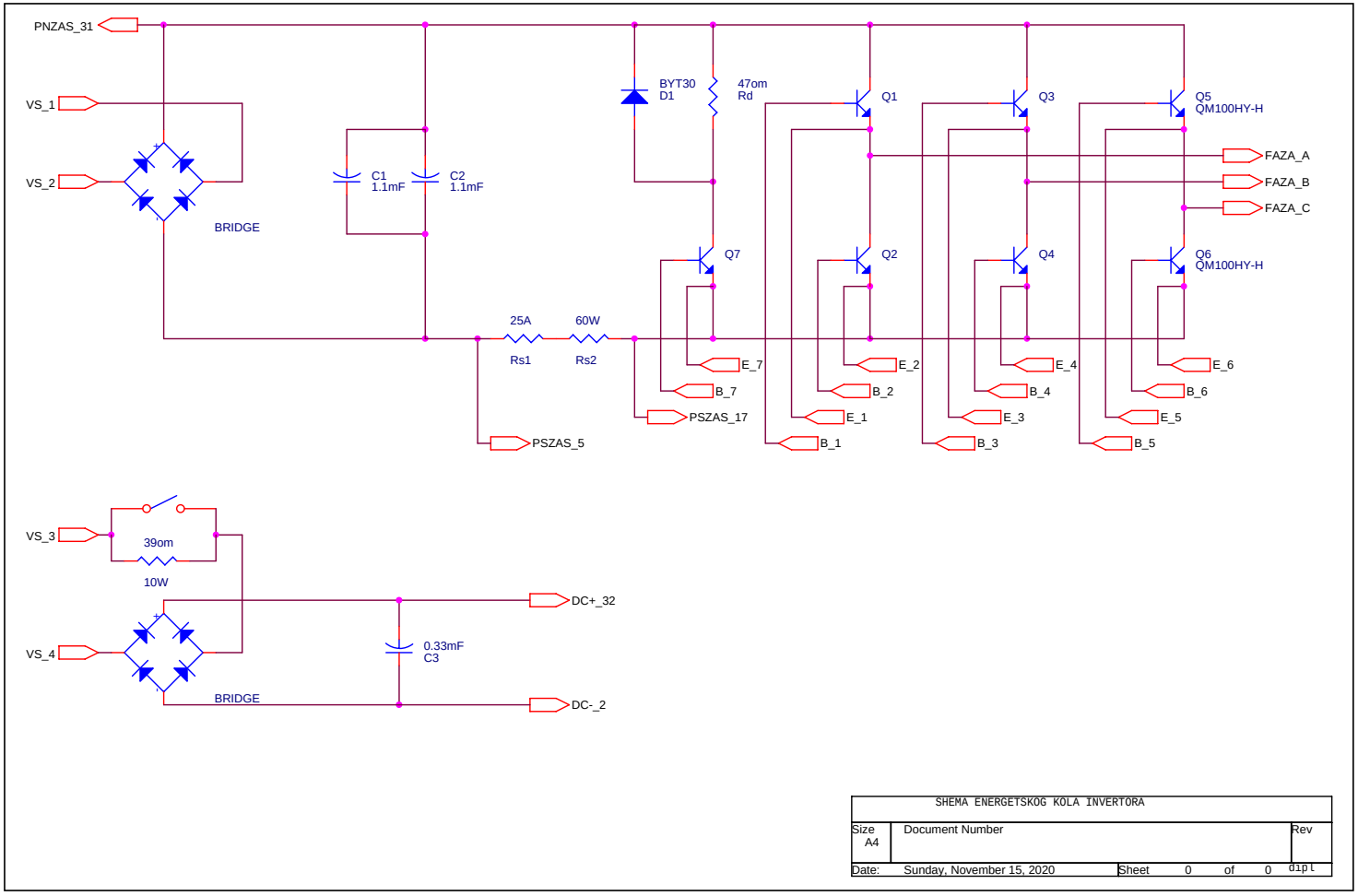
Slika 32. Električna šema veza kartice K7 (Kartica prekostrujne i prenaponske zaštite).



Slika 33. Električna šema veza kartice K8 (Kartica strujne regulacije).



Slika 34. Električna šema veza invertorskog mosta.



6.2. Izvorni kod S/W laboratorijske stanice VEKTRA

Listing1. Izvorni C kod programa *realtime.exe*

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <math.h>

typedef unsigned int UINT;

void interrupt (*old_timer_handler)();
void interrupt new_timer_handler();

int main(void)
{
   char str[16];

   /* Incijalizacija portova Prilagodne kartice 1 */
   outp(0x303,0x92);    /* 1PA-in,  1PB-in,   1PCup-in, 1PClo-out */
   outp(0x307,0x80);    /* 2PA-out, 2PB-out,     2PC-out */
   outp(0x30b,0x9a);    /* 3PA-in,  3PB-in,   3PC-out */

   /* Organizacija Real-Time prekidne rutine sa periodom T = 1 ms */
   disable();
   old_timer_handler = getvect(0x1c);

   outp(0x043, 0x36);   /* Novi sistemski tajmer (cntr #0 mode 3) */
   outp(0x040, 0xa8);   /* LSB za T = 1ms */
   outp(0x040, 0x04);   /* MSB za T = 1ms */

   setvect(0x1c, new_timer_handler);
   enable();

   /* Kontrolisani izlazak iz programa */
   while(1)
   {
      printf("> Enter QUIT to exit: ");
      gets(str);

      if(!(strcmp(str, "QUIT")))
      {
         disable();
         setvect(0x1c,old_timer_handler);

         outp(0x043,0x36);    /* Stari sistemski tajmer (cntr #0 mode 3) */
         outp(0x040,0xff);    /* Default LSB */
         outp(0x040,0xff);    /* Default MSB */
         enable();
         return(0);
      }
   }
}

void interrupt new_timer_handler()
{
   static UINT nTicTac = 0;

   /* Generisanje povorke pravougaonih impulsa */
   nTicTac++;
   nTicTac &= 0x0001;

   /* Prikaz generisane povorke na digitalnom osciloskopu */
```

```
outp(0x306, (nTicTac) ? 255 : 128);
}
```

Listing 2. Izvorni C kod programa *enc2000.exe*

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <math.h>

typedef unsigned int UINT;

void interrupt (*old_timer_handler)();
void interrupt new_timer_handler();

int main(void)
{
   char str[16];

   /* Incijalizacija portova Prilagodne kartice 1 */
   outp(0x303,0x92);    /* 1PA-in,  1PB-in,   1PCup-in, 1PClo-out */
   outp(0x307,0x80);    /* 2PA-out, 2PB-out,     2PC-out */
   outp(0x30b,0x9a);    /* 3PA-in,  3PB-in,   3PC-out */

   /* Organizacija Real-Time prekidne rutine sa periodom T = 1 ms */
   disable();
   old_timer_handler = getvect(0x1c);

   outp(0x043, 0x36);   /* Novi sistemski tajmer (cntr #0 mode 3) */
   outp(0x040, 0xa8);   /* LSB za T = 1ms */
   outp(0x040, 0x04);   /* MSB za T = 1ms */

   setvect(0x1c, new_timer_handler);
   enable();

   /* Kontrolisani izlazak iz programa */
   while(1)
   {
      printf("> Enter QUIT to exit: ");
      gets(str);

      if(!(strcmp(str, "QUIT")))
      {
         disable();
         setvect(0x1c,old_timer_handler);

         outp(0x043,0x36);    /* Stari sistemski tajmer (cntr #0 mode 3) */
         outp(0x040,0xff);    /* Default LSB */
         outp(0x040,0xff);    /* Default MSB */

         enable();

         return(0);
      }
   }
}


void interrupt new_timer_handler()
{
   static UINT nEncHiByte,
               nEncLoByte,
               nNewEnc,
```

```
                nOldEnc = 0;

    static int  nTeta = 0,
                nIncPos;

    /* Citanje dvosmernog brojaca enkoderskih impulsa */
    outp(0x302, 0x02);        /* Disable Encoder latch (output)   */
    nEncHiByte = inp(0x308);  /* Cita visih 4 bita 12-bit latch-a */
    nEncLoByte = inp(0x309);  /* Cita nizih 8 bita 12-bit latch-a */
    outp(0x302, 0x00);        /* Enable Encoder latch (output)    */

    nNewEnc  = nEncHiByte << 8;
    nNewEnc |= nEncLoByte;

    /* Odredjivanje inkrementa pozicije */
    nIncPos = nNewEnc - nOldEnc;
    nOldEnc = nNewEnc;

    if(nIncPos >  1250) nIncPos -= 2500;
    if(nIncPos < -1250) nIncPos += 2500;

    /* Odredjivanje pozicije vratila asinhronog motora */
    nTeta += nIncPos;

    if(nTeta > 1999) nTeta -= 2000;
    if(nTeta < 0)    nTeta += 2000;

    /* Prikaz detektovane pozicije na digitalnom osciloskopu */
    outp(0x306, (nTeta >> 5) + 128);
}
```

Listing 3. Izvorni C kod programa *sin2000.exe*

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <math.h>

typedef   unsigned int UINT;

#define   PI        3.1415927
#define   HIBYTE(w) (w >> 8)

void interrupt (*old_timer_handler)();
void interrupt new_timer_handler();

void fill_sine_table(int*);
int  SIN(const int*, const UINT);
int  COS(const int*, const UINT);

int  nSineTable[2000];

int main(void)
{
   char str[16];

   fill_sine_table(nSineTable);

   /* Incijalizacija portova Prilagodne kartice 1 */
   outp(0x303,0x92);    /* 1PA-in,  1PB-in,   1PCup-in, 1PClo-out */
   outp(0x307,0x80);    /* 2PA-out, 2PB-out,  2PC-out */
   outp(0x30b,0x9a);    /* 3PA-in,  3PB-in,   3PC-out */

   /* Organizacija Real-Time prekidne rutine sa periodom T = 1 ms */
```

```
    disable();
    old_timer_handler = getvect(0x1c);

    outp(0x043, 0x36);   /* Novi sistemski tajmer (cntr #0 mode 3) */
    outp(0x040, 0xa8);   /* LSB za T = 1ms */
    outp(0x040, 0x04);   /* MSB za T = 1ms */

    setvect(0x1c, new_timer_handler);
    enable();

    /* Kontrolisani izlazak iz programa */
    while(1)
    {
       printf("> Enter QUIT to exit: ");
       gets(str);

       if(!(strcmp(str, "QUIT")))
       {
          disable();
          setvect(0x1c,old_timer_handler);

          outp(0x043,0x36);    /* Stari sistemski tajmer (cntr #0 mode 3) */
          outp(0x040,0xff);    /* Default LSB */
          outp(0x040,0xff);    /* Default MSB */

          enable();
          return(0);
       }
    }
}

void interrupt new_timer_handler()
{
    static UINT   nEncHiByte,
                  nEncLoByte,
                  nNewEnc,
                  nOldEnc = 0;

    static int    nTeta,
                  nIncPos,
                  nSinTeta,
                  nCosTeta;

    /* Citanje dvosmernog brojaca enkoderskih impulsa */
    outp(0x302, 0x02);         /* Disable Encoder latch (output)   */
    nEncHiByte = inp(0x308);   /* Cita visih 4 bita 12-bit latch-a */
    nEncLoByte = inp(0x309);   /* Cita nizih 8 bita 12-bit latch-a */
    outp(0x302, 0x00);         /* Enable Encoder latch (output)    */

    nNewEnc  = nEncHiByte << 8;
    nNewEnc |= nEncLoByte;

    /* Odredjivanje inkrementa pozicije */
    nIncPos = nNewEnc - nOldEnc;
    nOldEnc = nNewEnc;

    if(nIncPos >  1250) nIncPos -= 2500;
    if(nIncPos < -1250) nIncPos += 2500;

    /* Odredjivanje pozicije vratila asinhronog motora */
    nTeta += nIncPos;

    if(nTeta > 1999) nTeta -= 2000;
    if(nTeta < 0)    nTeta += 2000;
```

```
   /* Generisanje obrtnih vektora */
   nSinTeta = SIN(nSineTable, nTeta);
   nCosTeta = COS(nSineTable, nTeta);

   /* Prikaz obrtnih vektora na digitalnom osciloskopu */
   outp(0x306, HIBYTE(nSinTeta) + 128);
}

void fill_sine_table(int *SineTable)
{
   register UINT  i;

   for(i = 0; i < 1000; i++)
   {
      SineTable[i] = (int)(32767*sin(2*PI*i/2000) + 0.5);
   }

   for(i = 1000; i < 2000; i++)
   {
      SineTable[i] = (int)(32767*sin(2*PI*i/2000) - 0.5);
   }
}

int SIN(const int *SineTable, const UINT Teta)
{
   UINT nTeta = Teta;

   return SineTable[nTeta];
}

int COS(const int *SineTable, const UINT Teta)
{
   UINT nTeta = Teta;

   nTeta += 500;

   if(nTeta >= 2000) nTeta -= 2000;

   return SineTable[nTeta];
}
```

Listing 4. Izvorni C kod programa *direct.exe*

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <dos.h>
#include <ctype.h>
#include <math.h>

#define   IDNOM        98
#define   IQNOM        114
#define   SLIPGAIN     2108
#define   LONG1999     131006464
#define   LONG2000     131072000
#define   IQMAX        119537664

#define   TRUE         1
#define   FALSE        0
#define   PI           3.1415927
typedef   unsigned int  UINT;
typedef   unsigned char BOOL;
```

```
union LONG {
       long ALL;

       struct {
          int LO;
          int HI;
       }  WORD;
};

void interrupt (*old_timer_handler)();
void interrupt new_timer_handler();

void fill_sine_table(int*);
int  SIN(const int*, const UINT);
int  COS(const int*, const UINT);

int nSineTable[2000];

int nIQref = 0;

int main(void)
{
   char str[16];

   /* Generisanje sinusne tabele */
   fill_sine_table(nSineTable);

   /* Incijalizacija portova Prilagodne kartice 1 */
   outp(0x303,0x92);    /* 1PA-in,  1PB-in,   1PCup-in, 1PClo-out */
   outp(0x307,0x80);    /* 2PA-out, 2PB-out,  2PC-out */
   outp(0x30b,0x9a);    /* 3PA-in,  3PB-in,   3PC-out */

   /* Organizacija Real-Time prekidne rutine sa periodom T = 1 ms */
   disable();
   old_timer_handler = getvect(0x1c);

   outp(0x043, 0x36);   /* Novi sistemski tajmer (cntr #0 mode 3) */
   outp(0x040, 0xa8);   /* LSB za T = 1ms */
   outp(0x040, 0x04);   /* MSB za T = 1ms */

   setvect(0x1c, new_timer_handler);
   enable();

   /* Kontrolisani izlazak iz programa i zadavanje Iq-referentne struje */
   while(1)
   {
      printf("> Iqref or QUIT to exit: ");
      gets(str);

         if(!(strcmp(str, "QUIT")))
         {
            disable();

            outp(0x304, 128);    /* Ialfa = 0 */
            outp(0x305, 128);    /* Ibeta = 0 */

            setvect(0x1c,old_timer_handler);

            outp(0x043,0x36);    /* Stari sistemski tajmer (cntr #0 mode 3) */
            outp(0x040,0xff);    /* Default LSB */
            outp(0x040,0xff);    /* Default MSB */

            enable();
            return(0);
```

```
          }
          nIQref = atoi(str);

          if(nIQref >  10)
          {
             nIQref =  10;
             printf("> Limit +10!\n");
          }

          if(nIQref < -10)
          {
             nIQref = -10;
             printf("> Limit -10!\n");
          }
       }
}

void interrupt new_timer_handler()
{
   static UINT  nEncHiByte,
                nEncLoByte,
                nNewEnc,
                nOldEnc = 0,
                nLoopID = 10;

   static int   nTetaE,
                nIncPos,
                nPosDif = 0,
                nSinTeta,
                nCosTeta,
                nIalfa,
                nIbeta;

   static long  nSlipInc;

   static union LONG ACC, TETA = {0};

   if(nLoopID == 10)
   {
      /* Prikaz detektovane brzine na digitalnom osciloskopu */
      outp(0x306, (nPosDif >> 2) + 128);
      nPosDif = 0;
   }

   if(--nLoopID == 0) nLoopID = 10;

   /* Citanje dvosmernog brojaca enkoderskih impulsa */
   outp(0x302, 0x02);         /* Disable Encoder latch (output)    */
   nEncHiByte = inp(0x308);   /* Cita visih 4 bita 12-bit latch-a    */
   nEncLoByte = inp(0x309);   /* Cita nizih 8 bita 12-bit latch-a    */
   outp(0x302, 0x00);         /* Enable Encoder latch (output)     */

   nNewEnc  = nEncHiByte << 8;
   nNewEnc |= nEncLoByte;

   /* Odredjivanje inkrementa pozicije */
   nIncPos = nNewEnc - nOldEnc;
   nOldEnc = nNewEnc;

   if(nIncPos >  1250) nIncPos -= 2500;
   if(nIncPos < -1250) nIncPos += 2500;

   /* Procena brzine obrtanja vratila asinhronog motora */
   nPosDif += nIncPos;
```

```
    /* Indirektno vektorsko upravljanje asinhronim motorom */
    TETA.WORD.HI += nIncPos;
    TETA.WORD.HI += nIncPos;

    nSlipInc = (long)SLIPGAIN*nIQref;

    TETA.ALL += nSlipInc;
    TETA.ALL += nSlipInc;

    if(TETA.ALL > LONG1999) TETA.ALL -= LONG2000;
    if(TETA.ALL < 0)        TETA.ALL += LONG2000;

    nTetaE = (UINT)TETA.WORD.HI;

    nSinTeta = SIN(nSineTable, nTetaE);
    nCosTeta = COS(nSineTable, nTetaE);

    ACC.ALL = (long)IDNOM*nCosTeta - (long)nIQref*nSinTeta;
    nIalfa = ACC.WORD.HI;  /* Referentna struja Ialfa */

    ACC.ALL = (long)IDNOM*nSinTeta + (long)nIQref*nCosTeta;
    nIbeta = ACC.WORD.HI;  /* Referentna struja Ibeta */

    outp(0x304, nIalfa + 128);
    outp(0x305, nIbeta + 128);
}

void fill_sine_table(int *SineTable)
{
    register UINT  i;

    for(i = 0; i < 1000; i++)
    {
        SineTable[i] = (int)(32767*sin(2*PI*i/2000) + 0.5);
    }

    for(i = 1000; i < 2000; i++)
    {
        SineTable[i] = (int)(32767*sin(2*PI*i/2000) - 0.5);
    }
}

int SIN(const int *SineTable, const UINT Teta)
{
    UINT nTeta = Teta;

    return SineTable[nTeta];
}

int COS(const int *SineTable, const UINT Teta)
{
    UINT nTeta = Teta;

    nTeta += 500;

    if(nTeta >= 2000) nTeta -= 2000;

    return SineTable[nTeta];
}
```

Listing 5. Izvorni C kod programa *speed.exe*

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <dos.h>
#include <ctype.h>
#include <math.h>

#define   IDNOM        98
#define   IQNOM        114
#define   SLIPGAIN     2108
#define   LONG1999     131006464
#define   LONG2000     131072000
#define   IQMAX        119537664

#define   TRUE         1
#define   FALSE        0

#define   PI           3.1415927

typedef   unsigned int  UINT;
typedef   unsigned char BOOL;

union LONG {
       long ALL;

       struct {
          int LO;
          int HI;
       }  WORD;
};

void interrupt (*old_timer_handler)();
void interrupt new_timer_handler();

void fill_sine_table(int*);
int  SIN(const int*, const UINT);
int  COS(const int*, const UINT);

int nSineTable[2000];

UINT   KP = 2393,    /* P-dejstvo regulatora brzine (vidi speed.mdl) */
       KI = 150;     /* I-dejstvo regulatora brzine (vidi speed.mdl) */

long   nSpeedRef = 0; /* Referentna brzina */

int main(void)
{
   char str[16];
   int  nRetVal;

   fill_sine_table(nSineTable);

   /* Incijalizacija portova Prilagodne kartice 1 */
   outp(0x303,0x92);    /* 1PA-in,  1PB-in,   1PCup-in, 1PClo-out */
   outp(0x307,0x80);    /* 2PA-out, 2PB-out,  2PC-out */
   outp(0x30b,0x9a);    /* 3PA-in,  3PB-in,   3PC-out */

   /* Organizacija Real-Time prekidne rutine sa periodom T = 1 ms */
   disable();
   old_timer_handler = getvect(0x1c);
```

```
   outp(0x043, 0x36);   /* Novi sistemski tajmer (cntr #0 mode 3) */
   outp(0x040, 0xa8);   /* LSB za T = 1ms */
   outp(0x040, 0x04);   /* MSB za T = 1ms */

   setvect(0x1c, new_timer_handler);
   enable();

   /* Kontrolisani izlazak iz programa i zadavanje referentne brzine */
   while(1)
   {
      printf("> Speed[rpm] or QUIT to exit: ");
      gets(str);

      if(!(strcmp(str, "QUIT")))
      {
         disable();

         outp(0x304, 128);      /* Ialfa = 0 */
         outp(0x305, 128);      /* Ibeta = 0 */

         setvect(0x1c,old_timer_handler);

         outp(0x043,0x36);    /* Stari sistemski tajmer (cntr #0 mode 3) */
         outp(0x040,0xff);    /* Default LSB */
         outp(0x040,0xff);    /* Default MSB */

         enable();
         return(0);
      }

      nRetVal = atoi(str);

      if(nRetVal >  800)
      {
         nRetVal =  800;
         printf("> Limit +800 [rpm]!\n");
      }

      if(nRetVal < -800)
      {
         nRetVal = -800;
         printf("> Limit -800 [rpm]!\n");
      }

      nSpeedRef = (long)100*nRetVal;
   }
}

void interrupt new_timer_handler()
{
   static UINT  nEncHiByte,
                nEncLoByte,
                nNewEnc,
                nOldEnc = 0,
                nLoopID = 10;

   static int   nTetaE,
                nIncPos,
                nPosDif = 0,
                nIQref,
                nSinTeta,
                nCosTeta,
                nIalfa,
                nIbeta;
```

```
static long  nSlipInc,
             nSpeedDif = 0,
             nSpeedFdb,
             nSpeedErr,
             nSpeedOld = 0,
             nOldAcc   = 0;

static union LONG ACC, TETA = {0};

if(nLoopID == 10)
{
   /* Prikaz regulisane brzine na digitalnom osciloskopu */
   outp(0x306, (nPosDif >> 2) + 128);

   /* Digitalna regulacija brzine (vidi SIMULINK model speed.mdl) */
   nSpeedFdb = (long)300*nPosDif;
   nPosDif = 0;

   nSpeedErr = nSpeedRef - nSpeedFdb;

   nSpeedDif = nSpeedFdb - nSpeedOld;
   nSpeedOld = nSpeedFdb;

   ACC.ALL = (long)nOldAcc + (long)KI*nSpeedErr - (long)KP*nSpeedDif;

   if(ACC.ALL > +IQMAX) ACC.ALL = +IQMAX;
   if(ACC.ALL < -IQMAX) ACC.ALL = -IQMAX;
   nOldAcc = ACC.ALL;

   nIQref  = ACC.WORD.HI >> 3;
}

if(--nLoopID == 0) nLoopID = 10;

/* Citanje dvosmernog brojaca enkoderskih impulsa */
outp(0x302, 0x02);        /* Disable Encoder latch (output)   */
nEncHiByte = inp(0x308);  /* Cita visih 4 bita 12-bit latch-a */
nEncLoByte = inp(0x309);  /* Cita nizih 8 bita 12-bit latch-a */
outp(0x302, 0x00);        /* Enable Encoder latch (output)    */

nNewEnc  = nEncHiByte << 8;
nNewEnc |= nEncLoByte;

/* Odredjivanje inkrementa pozicije */
nIncPos = nNewEnc - nOldEnc;
nOldEnc = nNewEnc;

if(nIncPos >  1250) nIncPos -= 2500;
if(nIncPos < -1250) nIncPos += 2500;

/* Procena brzine obrtanja vratila asinhronog motora */
nPosDif += nIncPos;

/* Indirektno vektorsko upravljanje asinhronim motorom */
TETA.WORD.HI += nIncPos;
TETA.WORD.HI += nIncPos;

nSlipInc = (long)SLIPGAIN*nIQref;

TETA.ALL += nSlipInc;
TETA.ALL += nSlipInc;

if(TETA.ALL > LONG1999) TETA.ALL -= LONG2000;
if(TETA.ALL < 0)        TETA.ALL += LONG2000;
```

```
    nTetaE = (UINT)TETA.WORD.HI;

    nSinTeta = SIN(nSineTable, nTetaE);
    nCosTeta = COS(nSineTable, nTetaE);

    ACC.ALL = (long)IDNOM*nCosTeta - (long)nIQref*nSinTeta;
    nIalfa = ACC.WORD.HI;  /* Referentna struja Ialfa */

    ACC.ALL = (long)IDNOM*nSinTeta + (long)nIQref*nCosTeta;
    nIbeta = ACC.WORD.HI;  /* Referentna struja Ibeta */

    outp(0x304, nIalfa + 128);
    outp(0x305, nIbeta + 128);
}

void fill_sine_table(int *SineTable)
{
    register UINT  i;

    for(i = 0; i < 1000; i++)
    {
        SineTable[i] = (int)(32767*sin(2*PI*i/2000) + 0.5);
    }

    for(i = 1000; i < 2000; i++)
    {
        SineTable[i] = (int)(32767*sin(2*PI*i/2000) - 0.5);
    }
}

int SIN(const int *SineTable, const UINT Teta)
{
    UINT nTeta = Teta;

    return SineTable[nTeta];
}

int COS(const int *SineTable, const UINT Teta)
{
    UINT nTeta = Teta;

    nTeta += 500;

    if(nTeta >= 2000) nTeta -= 2000;

    return SineTable[nTeta];
}
```

6.3 *Simulink* model brzinske regulacione petlje *speed.mdl*

Simulink model brzinske regulacione petlje u laboratorijskoj stanici VEKTRA *speed.mdl* izveden je iz SIMULINK modela *A.mdl*, koji je detaljno opisan u okviru laboratorijske vežbe: *"Ispitivanje dinamičkih svojstava digitalno regulisanog brzinskog servomehanizma putem računarskih simulacije"*.

Pokreće se izvršenjem komandne datoteke *go_speed.m* iz komandne linije MATLAB-a:

go_speed <Enter>

Napomena: SIMULINK model *speed.mdl* i komandna datoteka *go_speed.m* moraju biti na stazi vidljivoj MATLAB-u.

6.4 Opis programa *vektor.exe*

Program za indirektno vektorsko upravljanje asinhronim motorom sa grafičkim interfejsom *vektor.exe* nalazi se u direktorijumu D:\TC\VEKTRA, na PC računaru laboratorijske stanice VEKTRA. Poseduje grafički interfejs koji omogućava posmatranje talasnih oblika momenta, brzine i pozicije u realnom vremenu.

Pokreće se iz komandne linije DOS-a naredbom:

D:\TC\VEKTRA\vektor.exe <Enter>

Po pokretanju programa, aktivan je PI regulator brzine. Pomoću opcije SEMAFOR, moguće je izabrati nelinearni PID regulator pozicije ili, direktno zadavanje momenta. Zadavanjem vrednosti SEMAFOR = 1, aktivira se PI regulator brzine (*default* vrednost); zadavanjem vrednosti SEMAFOR = 2, aktivira se nelinearni PID regulator pozicije; zadavanjem vrednosti SEMAFOR = 3, direktno se zadaje vrednost momenta asinhronog motora.

U toku rada, moguće je menjati parametre regulatora brzine (KP i KI), pozicije (KP, KI i KD) i algoritma indirektnog vektorskog upravljanja (SLIPGAIN) i zadavati vrednost i trenutak delovanja odkočne pobude (momenta, brzine i pozicije).

Talasni oblici momenta, brzine i pozicije koji se ispisuju na monitoru PC računara, mogu se sačuvati u vidu ASCII datoteke sa ekstenzijom *.prn. Najpre se pomoću opcije FREEZ, zaustavi dalja promena talasnih oblika na ekranu; pomoću opcije ZOOM, naznači se i uveća deo ekrana od interesa a zatim se, izborom opcije SAVE SCREEN, taj deo čuva u tekstualni fajl sa ekstenzijom *.prn. Dobijeni fajl se nalazi u direktorijumu D:\TC\VEKTRA.

7. Literatura

[1] Slobodan N. Vukosavić: *"Digitalno upravljanje električnim pogonima ", Akademska misao, ETF Beograd, 2003.*

[2] Slobodan N. Vukosavić: *"Predavanja iz predmeta Mikroprocesorsko upravljanje elektromotornim pogonima ",* Elektrotehnički fakultet, Beograd, 1996.