



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ



Драган Радановић

**Једно решење програмске подршке
система за векторско управљање
мотором једносмерне струје без
четкица**

ДИПЛОМСКИ РАД
- Основне академске студије -

Нови Сад, 2014



КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, РБР:			
Идентификациони број, ИБР:			
Тип документације, ТД:	Монографска документација		
Тип записа, ТЗ:	Текстуални штампани материјал		
Врста рада, ВР:	Завршни (Bachelor) рад		
Аутор, АУ:	Драган Радановић		
Ментор, МН:	доц. Пјевалица Небојша		
Наслов рада, НР:	Једно решење програмске подршке система за векторско управљање мотором једносмерне струје без четкица		
Језик публикације, ЈП:	Српски / латиница		
Језик извода, ЈИ:	Српски		
Земља публиковања, ЗП:	Република Србија		
Уже географско подручје, УГП:	Војводина		
Година, ГО:	2014		
Издавач, ИЗ:	Ауторски репринт		
Место и адреса, МА:	Нови Сад; трг Доситеја Обрадовића 6		
Физички опис рада, ФО: (поглавља/страна/цитата/табела/слика/графика/прилога)	7/38/1/5/12/0/0		
Научна област, НО:	Електротехника и рачунарство		
Научна дисциплина, НД:	Рачунарска техника		
Предметна одредница/Кључне речи, ПО:	UART(SCI) magistrala,,F28035 процесор, MFC		
УДК			
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад		
Важна напомена, ВН:			
Извод, ИЗ:	У раду је приказано једно решење реализације програмске подршке за управљање мотором једносмерне струје без четкица ,користећи Texas Instruments-ов комплет "TMDS2MTRPFCKIT", као и реализација графичке корисничке спрете за управљање мотором путем сервиске асинхроне комуникације.		
Датум прихватања теме, ДП:			
Датум одбране, ДО:			
Чланови комисије, КО:	Председник:	др Милош Сланкаменац	
	Члан:	др Илија Башићевић	Потпис ментора
	Члан, ментор:	др Небојша Пјевалица	



KEY WORDS DOCUMENTATION

Accession number, ANO:		
Identification number, INO:		
Document type, DT:	Monographic publication	
Type of record, TR:	Textual printed material	
Contents code, CC:	Bachelor Thesis	
Author, AU:	Dragan Radanović	
Mentor, MN:	doc. Nebojša Pjevalica	
Title, TI:	One solution softwer support (firmware) for vector control of brushless DC motor	
Language of text, LT:	Serbian	
Language of abstract, LA:	Serbian	
Country of publication, CP:	Republic of Serbia	
Locality of publication, LP:	Vojvodina	
Publication year, PY:	2014	
Publisher, PB:	Author's reprint	
Publication place, PP:	Novi Sad, Dositeja Obradovica sq. 6	
Physical description, PD: (chapters/pages/ref./tables/pictures/graphs/appendices)	7/38/1/5/12/0/0	
Scientific field, SF:	Electrical Engineering	
Scientific discipline, SD:	Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, S/KW:	UART(SCI) bus, F28035 chip, MFC	
UC		
Holding data, HD:	The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, N:		
Abstract, AB:	The paper explains the detail of software support (firmware) for vector control of BLDC motors, using Texas Instruments "TMDS2MTRPCKIT" kit, and graphical user interface for motor control via serial asynchronous communication.	
Accepted by the Scientific Board on, ASB:		
Defended on, DE:		
Defended Board, DB:	President: Miloš Slankamenac, PhD	
Member:	Ilija Bašićević, PhD	Menthor's sign
Member, Mentor:	Nebojša Pjevalica, PhD	

SADRŽAJ

1.	Uvod.....	2
1.1	Zadatak rada	3
2.	Teorijske osnove	4
2.1	Vektorska kontrola.....	4
2.2	Klarkinia transformacija.....	5
2.3	Parkova transformacija.....	6
2.4	Impulsno širinski generator.....	7
2.5	Procena pozicije rotora.....	7
3.	Texas Instruments-ovo rešenje za vektorsko upravljanje	9
3.1	Kratak pregled Texas Instruments-ovog rešenja programske podrške za vektorsko upravljanje motorom bez četkica bez senzora.....	9
3.2	Impulsno širinska modulacija	10
3.3	Analogno digitalna konverzija	12
3.4	Parkova transformacija.....	13
3.5	Inverzna Parkova transformacija.....	13
3.6	Klarkina transformacija.....	14
3.7	Prostorno vektorski modulator	14
3.8	Proporcionalno integralni diferencionalni regulator	15
3.9	Kontroler „ramp“ signala	16
3.10	Generator „ramp“ signala	16
3.11	Kalkulator faznog napona.....	17
3.12	Estimator ugla rotora	17
3.13	Estimator brzine rotora	18

4.	Programsko rešenje.....	19
4.1	Programska podrška	19
4.1.1	2xPM_Motors modul	19
4.1.2	Asinhrona seriska komunikacija	20
4.2	Grafička korisnička sprega.....	24
5.	Ispitivanje i verifikacija	26
5.1	Ispitivanje i analiza programske podrške	27
5.2	Ispitivanje i verifikacija grafičke korisničke sprege	28
6.	Zaključak	30
7.	Literatura.....	32

SPISAK SLIKA

Slika 2.1 Klarkina transformacija	5
Slika 2.2 Klarkina transformacija u vremenskom domenu	5
Slika 2.3 Parkova transformacija	6
Slika 2.4 Parkova transformacija u vremenskom domenu.....	6
Slika 3.1 Dijagram modula programske podrške [14]	9
Slika 3.2 Blok dijagram podmodula impulsno širinskog modulatora	11
Slika 3.3 Blok dijagram analogno digitalnog konvertora	12
Slika 4.1 Blok dijagram modula za serijsku komunikaciju	21
Slika 4.2 Dijagram prekida serijske komunikacije	21
Slika 4.3 Izgled grafičke korisničke sprege	24
Slika 5.1 Razvojna ploča Multi-Axis DMC.....	26
Slika 5.2 Grafički prikaz zauzetosti procesora u μ s.....	28

SPISAK TABELA

Tabela 3.1 Objasnjene modula programske podrške	10
Tabela 4.1 Funkcije programske podrške	22
Tabela 4.2 Spisak komandi	23
Tabela 4.3 Funkcije grafičke korisničke sprege.....	25
Tabela 5.1 Analiza zauzetosti procesora pri izvršavanju blokova programske podrške za vektorsku kontrolu	27

SKRAĆENICE

CPU	- <i>Central Processor Unit</i> , Centralni procesor
GND	- Oznaka za signal na nultom potencijalu
UART	- <i>Universal asynchronous receiver/transmitter</i> , Univerzalni asinhroni prijemnik/predajnik
SCI	- <i>Serial Communication Interface</i> , Serijska komunikaciona sprega
FOC	- <i>Field-Oriented Control</i> , vektorska kontrola
PWM	- <i>Pulse-width modulation</i> , impulsno širinski generator
BLDC	- <i>Brushless Direct Current motor</i> , Motor jednosmerne struje bez četkica
GUI	- <i>Graphical User Interface</i> , Grafička korisnička sprega
PID	- <i>Proportional integral derivative controller</i> , Proporcionalno-integralni-diferencijalni kontroler
DTC	- <i>Direct Torque Control</i> , Direktna kontrola obrtnog momenta
ADC	- <i>Analog-Digital Convertor</i> , Analogno-digitalni konvertor
ISR	- <i>Interrupt Service Routine</i> , Prekidna servisna rutina
TI	- <i>Texas Instruments</i>
GPIO	- <i>General-purpose input/output</i> , Opštenamenenski ulaz-izlaz
NRZ	- <i>Non-Return to zero</i> , Bet povratka na nulti potencijal
RPM	- <i>Round per minute</i> , Broj obrtaja u minuti
MFC	- <i>Microsoft Foundation Class</i> ,
JTAG	- <i>Joint Test Action Group</i> ,
SoC	- <i>Start-of-Conversions</i> , Početak AD konverzije

1. Uvod

Vektorsku kontrolu motora su razvili K. Hase (K.Hasse) sa Tehničkog Univerziteta u Darmštatu i F. Blaške (F. Blaschke) iz Siemens AG kompanije. Vektorska kontrola predstavlja jedan od važnijih izuma u oblasti upravljanja motorima naizmenične struje [1]. Koncept vektorske kontrole podrazumeva da se upravlja fazom (komponenta fluksa) i kvadraturom (komponenta obrtnog momenta). Osnovni koncept indirektne vektorske kontrole bez merenja fluksa je predložio Hasse 1968 [2], a direktnu vektorskiju kontrolu, koja koristi direktno merenje fluksa da bi odredila stvarnu poziciju rotora, je razvijeno Blaschke 1971 [3]. Vektorska kontrola je u potpunosti razvijena tek 1980-tih godina u Japanu, pojavom sofisticirane digitalne kontrolne, koristeći upravljačke jedinice kao što su 32-bitni mikroprocesori.

U elektromotornim pogonima tzv. srednjih i visokih performansi zahteva se odvojeno upravljanje fluksom i obrtnim momentom motora, tj. nezavisno upravljanje brzinom i momentom. Motori jednosmerne struje (sa nezavisnom pobudom) konstrukcionalno imaju odvojeno upravljanje fluksom i obrtnim momentom preko struja rotora i struja statora. Zbog toga su, i pored visoke cene i složenog održavanja, u dugom vremenskom periodu dominirali u pogonima promenljive brzine. Sa razvojem jeftinih pretvarača energetske elektronike i digitalnih mikroprocesora, razvijeni su algoritmi upravljanja mašinama naizmenične struje. Tako su Teslini trofazni asinhroni motori, mašine jednostavne konstrukcije i niske cene, dobrih pogonskih karakteristika, i jednostavnog održavanja, počeli da se koriste i u pogonima visokih performansi. Današnji trend razvoja je takav da u savremenim elektromotornim pogonima asinhronе mašine polako potiskuju mašine jednosmerne struje. [4]

“Veoma brz napredak tehnologije u oblasti izrade integrisanih kola veoma velike skale integracije, kao i tehnika automatizacije razvoja elektronskih kola doveli su do stvaranja

mogućnosti za razvoj složenih i istovremeno malih (kompaktnih) upravljačkih komponenti za industrijske sisteme koji po svojim osobinama zadovoljavaju najviše standarde .

Motori jednosmerne struje sa stalnim magnetima bez četkica mogu se posmatrati kao neka vrsta trofaznog sinhronog motora, koji ima stalne magnete na rotoru, a koji nema mehanički komutator i četkice. Komutacija se postiže elektronskim prekidačima, koji snabdevaju strujom namotaje motora u sinhronizaciji sa pozicijom rotora.” [19]

1.1 Zadatak rada

Analizirati Texas Instrumets-ovo rešenje vektorske kontrole motora jednosmerne struje bez četkica (BLDC motora) urađeno na platformi “TMDS2MTRPFCKIT”. Detaljniju karakterizaciju dati u sledećim pravcima:

1. Analiza zauzetosti procesora pri izvršavanju blokova programske podrške za vektorskiju kontrolu.
2. Analiza akvizicionih performansi platforme (mogucnost za precizno podešavanje vremenskog generatora impulsa koji rukuju A/D konvertorom). Ispitivanje impulsno širinskog generatora (PWM generatora).
3. Realizovati UART spregu (na platformi, kao i u korisničkoj aplikaciji) putem koje se izdaju komande platformi da obavlja osnovne operacije: pokretanje, podešavanje brzine, zaustavljanje motora, praćenje faznih struja, i brzine motora.
4. Realizovati grafičku korisničku spregu koja sa strane personalnog racunara na zahtev korisnika upravlja platformom i omogućava vizuelizaciju stanja sistema.

2. Teorijske osnove

Upravljanje motorima jednosmerne struje svodi se na upravljanje naponom. Brzina obrtanja motora direktno zavisi od napona, a obrtni moment od struje. Motori naizmenične struje mogu biti sinhroni ili asinhroni. Sinhroni motori se kreću vrlo tačnom brzinom, u skladu sa frekvencijom napajanja. Asinhroni motori su mašine jednostavne konstrukcije, jeftini su i pouzdani u radu. Brzina rada je promenjiva, u zavisnosti od opterećenja, ali je uglavnom nešto niža od količnika frekvencije mreže i broja para polova. Postoje i univerzanli motori koji rade i na jednosmernu i na naizmenični struju. Univerzalni motor je motor sa komutatorom i četkicama, sličan motoru jednosmerne struje.

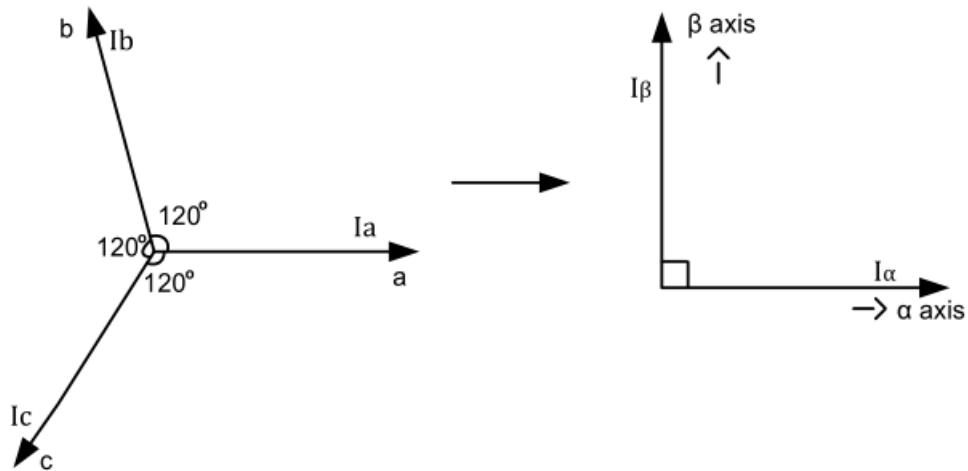
Kontrola motora naizmenične struje sa nepromenjivom brzinom se svodi na motore koji se direkno povezuju na napon napajanja, i na motore sa starterima. Kod kontrole motora naizmenične struje sa promenjivom brzinom koriste se različiti invertori, pretvarači frekvencije i elektronski komutatori. [5][6]

2.1 Vektorska kontrola

Vektorska kontrola je metod upravljanja sinhronim i asihronim motorima naizmenične struje. Maksimalan obrtni moment se javlja kad je ugao izmedju polja statora i polja rotora jednak 90 stepeni. Cilj je orijetisati vektor struje statora tako da bude pod pravim uglom u odnosu na vektor rotora. Ako je rotor napravljen od stalnih magneta, onda je potrebno orijetisati samo polje generisano namotajima u statoru tako da ugao izmedju njih iznosi 90 stepeni. Frekvencija napajanja je konstantna, kontroliše se samo amplituda, najčešće sa impulsno širinskim generatorom. Ako znamo poziciju rotora, upravljanje motora se svodi na izracunavanje struje po fazama da bi se dobilo polje orijentisano pod pravim uglom u odnosu na polje rotora, tj maksimalni obrtni momenat.

2.2 Klarkina transformacija

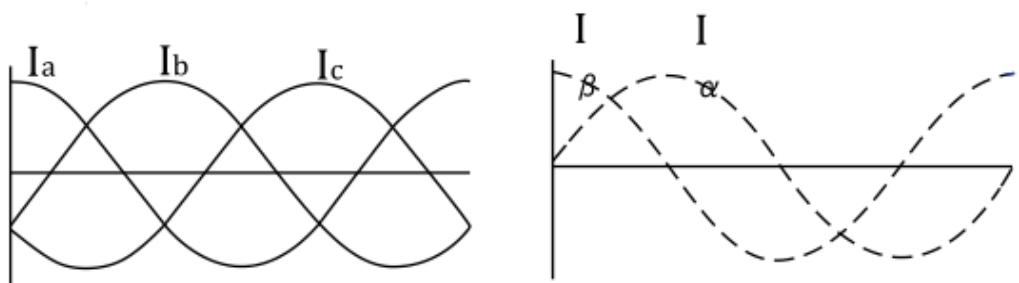
Klarkina transformacija pretvara vrednosti balansiranog trofaznog sistema u vrednosti balansiranog dvofaznog sistema. [6] Tri faze možemo predstaviti kao jedan vektor, koji je jednak vektorskom zbiru faza. Taj vektor možemo predstaviti sa dve promenjive, i_α i i_β . Na taj način smanjujemo broj promenjivih kojima upravljam.



Slika 2.1 Klarkina transformacija

$$i_\alpha = i_A$$

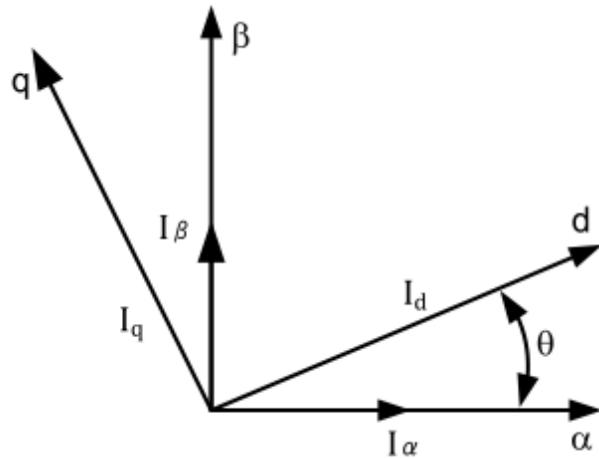
$$i_\beta = \frac{1}{\sqrt{3}} i_B + \frac{2}{\sqrt{3}} i_b$$



Slika 2.2 Klarkina transformacija u vremenskom domenu

2.3 Parkova transformacija

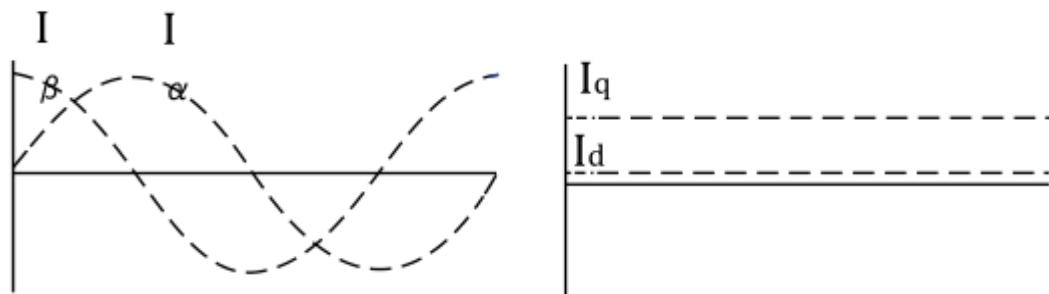
Parkova transformacija pretvara vektor, iz balansiraniranog dvofaznog pravouglog stacionarnog sistema u balansirani pravougli rotirajući sistem. [7] (Jednostavnije je posmatrati rotirajuci sistem sa rotirajuceg mesta.)



Slika 2.3 Parkova transformacija

$$i_d = i_\alpha \cos \theta_d + i_\beta \sin \theta_d$$

$$i_q = -i_\alpha \cos \theta_d + i_\beta \cos \theta_d$$



Slika 2.4 Parkova transformacija u vremenskom domenu

2.4 Impulsno širinski generator

Impulsno širinska modulacija je tehnicka modulacija impulsa, koja menja širinu, odnosno vremensko trajanje impulsa. Ovu tehniku je moguće koristiti i za prenos signala, ali se češće koristi u pretvaračima napona. Srednja vrednost napona i struje koja se dovodi na izlaz impulsno širinskog modulatora se kontroliše brzim prekidanjem izmedju napona napajanja i mase. Što je duze prekidač u propusnom stanju, to je na izlazu veći napon.

Frekvencija na kojoj se prekidanje odvija mora biti veća nego što uredjaj koji se napaja može da primeti. Impulsno širinska modulacija se koristi u električnim šporetima, svetlosnim dimmerima, za napajanje električnih motora, kao i za audio pojačavače, i računarska napajanja.

Faktor ispune opisuje koliko dugo je prekidač u propusnom (otvorenom) položaju u odnosu na to koliko je u nepropusnom (zatvorenom) položaju. Faktor ispune se izražava se procentima, gde 100% označava da je prekidač stalno u propusnom položaju.

Glavna prednost napajanja na osnovu impulsno širinskog modulatora su jako mali gubitci. Kada je prekidačko kolo isključeno, praktično struja ne teče, a kada je u prekidačko kolo uključeno skoro da nema pada napona na prekidačkom kolu, pa su u oba slučaja gubitci bliski nuli. (Manu mu je to što može da unese šum(smetnje) u kolo.) [8]

2.5 Procena pozicije rotora

Za ispravnu vektorskiju kontrolu motora sa stalnim magnetima bez četkica neophodno je znati tačnu poziciju rotora, da bi mogli proizvesti odgovarajuću pobudu statora. Postoji dva generalna pristupa za dobijanje informacije o poziciji rotora: sa senzorima i bez senzora.

Senzorska kontrola se često koristi u primenama gde je potreban veliki startni obrtni momenat, ili gde je početni obrtni momenat promenljiv. Samo sa tačnom informacijom o poziciji rotora moguće je optimalna kontrola, i korektno pobudjivanje faza statora. Motori sa senzorima uvek znaju tačnu poziciju, čak i pri niskim brzinama, ili kad su u stanju mirovanja. Ovo omogućava da se motori bez četkica ponašaju kao motori sa četkicama, i da isporuče maksimalan obrtni momenat pri nultoj brzini, bez gubitaka na četkicama i komutatoru, koji su karakteristični za motore sa četkicama. Koriste se Holovi senzori (*Hall effect sensor*). Današnji mikroprocesori rade na mnogo većim frekvencijama radnog takta, nego sto su frekvencije rada motora, tako da se bez problema mogu pokupiti informacije o poziciji, obraditi, i dovesti na faze statora odgovarajuće pobude.

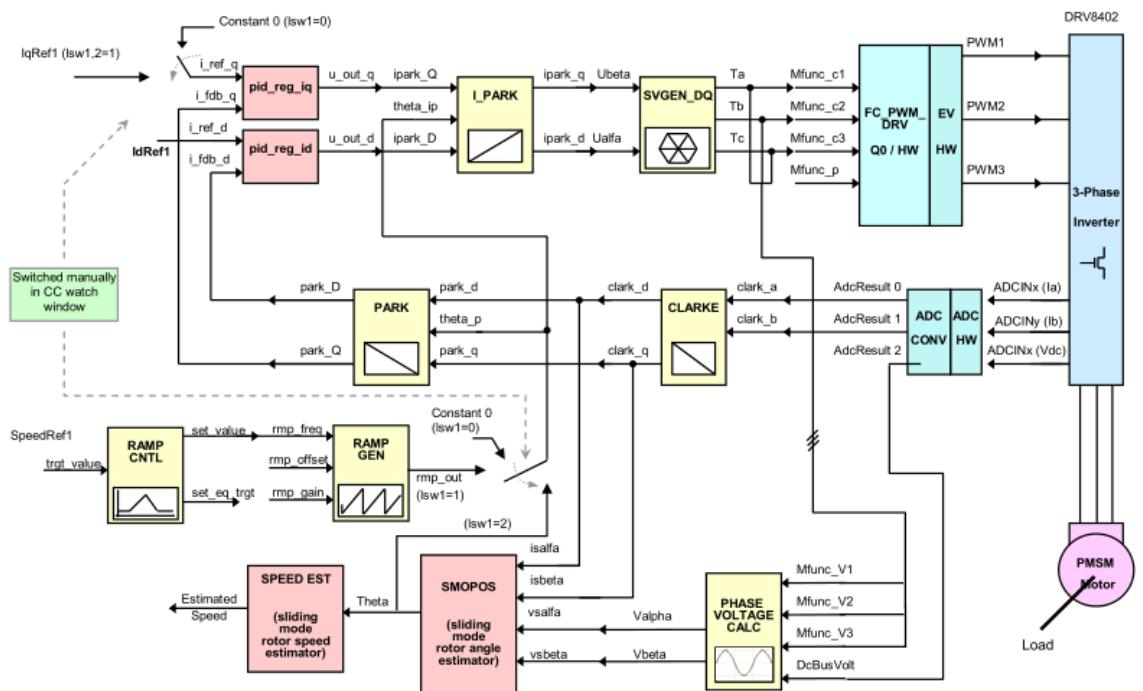
Kontrola bez senzora, sa druge strane, nema informaciju o poziciji rotora sve dok se motor već ne vrti nekom određenom brzinom, tj dok ne generiše dovoljno povratne elektromotorne sile

(*back-EMF*) za generisanje procene pozicije rotora. Potrebna je posebna rutina za pokretanje motora, pre prelaska na vektorsku kontrolu. Za pokretanje najčešće se koristi "*ramp-up*" metod, koji se sastoji od dovodjenja pobude na dve faze, tako da se rotor dovede u određenu poziciju ("parkiranje"), a posle toga se dovodi određeni šablon pobuda, da bi se motor postepeno ubrzao do brzine na kojoj se generiše dovoljno povratne elektromotorne sile. Kad se dosigne željena brzina, odnosno kada se generiše dovoljno povratne elektro motorne sile, algoritam upravljanja prebacuje se na vektorsklu kontrolu. Motori sa senzorima se čine superiornijim u odnosu na motore bez senzora, motori bez senzora su jeftiniji (nemaju senzore) i jednostavniji za održavanje (ne kvare se senzori).

U zavisnosti od primene biraju se odgovarajući motori. Za primene gde nije bitan obrtni momenat pri nultoj brzini, biraju se motori bez senzora. Ako su zahtevi strožiji, onda se koriste motori sa senzorima.

3. Texas Instruments-ovo rešenje za vektorsko upravljanje

3.1 Kratak pregled Texas Instruments-ovog rešenja programske podrške za vektorsko upravljanje motorom bez četkica bez senzora



Slika 3.1 Dijagram modula programske podrške [14]

Naziv modula	Objašnjenje
I_PARK	Inverzna Parkova transformacija
SVGEN_DQ	Prostorni vektorski modulator (sa inverznom Klarkinom transformacijom)
PARK	Direktna Parkova transformacija
CLARKE	Dirketna Klarkina transformacija
PID	PID kontroler
RAMP_CNTL	Kontroler ramp signala(uzlazni signal)
RAMP_GEN	Generator ramp signala
PHASE_VOLT_CALC	Kalkulator faznog napona
SMPOS	Estimator ugla rotora
SPEED_EST	Estimator brzine rotora
ADC_CONV	Analogno-digitalni konvertor
PWM_DRV	Generator impulsno-širinsko modulisanog signala

Tabela 3.1 Objasnjene modula programske podrške

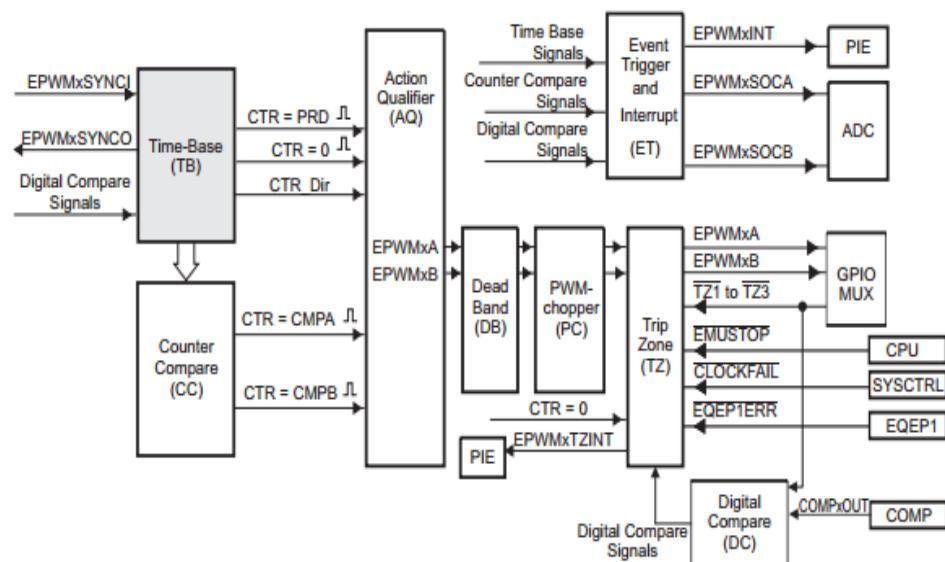
Rešenje koje je koristio TI je "standardna" (često korišćena) vektorska kontrola motora bez senzora. [12] Na ulaz Klarkine transformacije dovode se vrednosti strujnih odabiraka sa ADC-a. Dovoljno je dovesti samo odbirke sa dve faze, zbog pretpostavke da je vektorski zbir sve tri faze jednak nuli. Inverzna Parkova transformacija se odvija u bloku za prostorno vektorskiju modulaciju(SVGEN_DQ). U osnovnom rešenju prelaz izmedju otvorene petlje (ramp pobuda) i zatvorene petlje (vektorska kontrola) se ručno odvija.

3.2 Impulsno širinska modulacija

Za efikasnu impulsno širinsku modulaciju potrebno je generisati kompleksan impulsno-širinski talasni oblik, poželjno sa minimalnim procesorskim opterećenjem. Modul za generisanje PWM signala koji je korišćen u TI rešenju je programabilan, fleksibilan i lak za korišćenje.[17]

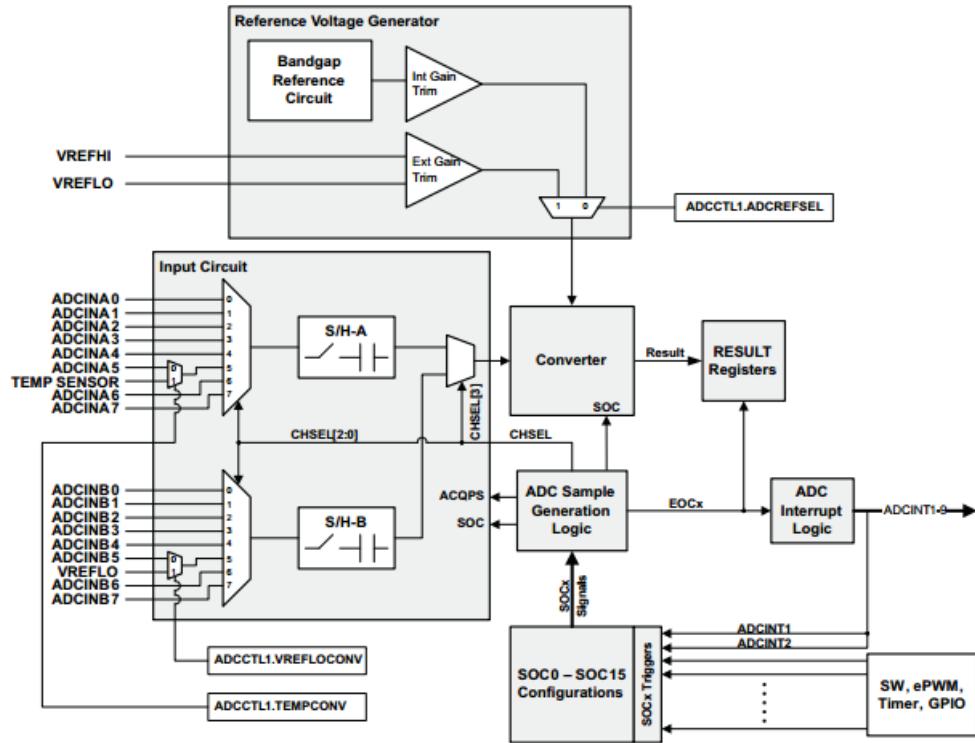
Svaki PWM modul podržava sledeće mogućnosti:

- Asinhrona kontrola PWM putem programske podrške
- Programabilna fazna kontrola za kontrolu kašnjenja u zavisnosti na druge PWM module
- Utiskivanje “mrtvog vremena” u PWM signal, sa posebnom kontrolom kašnjena uzlazne i silazne ivice PWM signala.
- Za izlaz se može postaviti visok nivo, nizak nivo, ili u stanje visoke impedanse.
- Može da generiše sistemski prekid, ili da pošalje signal analogno digitalnom kovertoru za početak konverzije.
- Programabilni dogadjaji smanjuju opterećenje centralnog procesora.



Slika 3.2 Blok dijagram podmodula impulsno širinskog modulatora

3.3 Analogno digitalna konverzija



Slika 3.3 Blok dijagram analogno digitalnog konvertora

Jezgro ADC je jedan 12-bitni ADC kovertor sa dva odabiračka (*sample-and-hold*) kola, koja omogucavaju konverziju dva istovremena signala. Na ta dva kola su dovedeni multiplekseri, na koje se mogu povezati do 16 razlicitih ulaza. [16]

Osobine modula:

- 12-bitno jezgro sa dva ugradjena odabiračka kola
- Simultani ili sekvencijalni načini odabiranja.
- Radi na sistemskom taktu, bez potrebe preskaliranja.
- Do 16 kanala, multipleksirani ulazi
- 16 registara za smeštanje rezultata
- Više okidača početka konverzije
 - Iz programske podrške
 - Sa impulsno širinskog generatora
 - Sa opštenamenskih ulaza/izlaza
 - Sa sistemskog tajmera

3.4 Parkova transformacija

Modul Parkove transformacije je realizovan kao struktura. Deklaracija strukture, direktno preuzeta iz projektne datoteke park.h data je u tekstu niže.

```
typedef struct
{
    _iq Alpha;           // Input: stationary d-axis stator variable
    _iq Beta;            // Input: stationary q-axis stator variable
    _iq Angle;           // Input: rotating angle (pu)
    _iq Ds;              // Output: rotating d-axis stator variable
    _iq Qs;              // Output: rotating q-axis stator variable
    void (*calc)();      // Pointer to calculation function
} PARK;

typedef PARK *PARK_handle;
```

3.5 Inverzna Parkova transformacija

Modul inverzne Parkove transformacije je realizovan kao struktura. Deklaracija strukture, direktno preuzeta iz projektne datoteke ipark.h data je u tekstu niže.

```
typedef struct
{
    _iq Alpha;           // Output: stationary d-axis stator variable
    _iq Beta;            // Output: stationary q-axis stator variable
    _iq Angle;           // Input: rotating angle (pu)
    _iq Ds;              // Input: rotating d-axis stator variable
    _iq Qs;              // Input: rotating q-axis stator variable
    void (*calc)();      // Pointer to calculation function
} IPARK;

typedef IPARK *IPARK_handle;
```

3.6 Klarkina transformacija

Modul CLARKE realizuje Klarikinu transformaciju. Realizovan je kao struktura. Pozivanjem funkcije izračunavaju se vrednosti Klarkine transformacije. Na ulaz se dovode dve faze, zbog prepostavke da je sistem balansiran, odnosno da je zbir sve tri faze jednak nuli. Deklaracija strukture, direktno preuzeta iz projektne datoteke clarke.h data je u tekstu niže.

```
typedef struct
{
    _iq As;                      // Input: phase-a stator variable
    _iq Bs;                      // Input: phase-b stator variable
    _iq Alpha;                   // Output: stationary d-axis stator variable
    _iq Beta;                    // Output: stationary q-axis stator variable
    void (*calc)();              // Pointer to calculation function
} CLARKE;

typedef CLARKE *CLARKE_handle;
```

3.7 Prostorno vektorski modulator

Prostorno vektorska modulacija je algoritam za kontrolu impulsno širinske modulacije. [21] Najčešće se koristi za pogon motora naizmenične struje promenjivim brzinama koristeći jednosmernu struju. Prostorno vektorski modulator je realizovan kao struktura. Deklaracija strukture, direktno preuzeta iz projektne datoteke svgen_dq.h data je u tekstu niže.

```
typedef struct
{
    _iq Ualpha;                  // Input: reference alpha-axis phase voltage
    _iq Ubeta;                  // Input: reference beta-axis phase voltage
    _iq Ta;                     // Output: reference phase-a switching function
    _iq Tb;                     // Output: reference phase-b switching function
    _iq Tc;                     // Output: reference phase-c switching function
    void (*calc)();              // Pointer to calculation function
} SVGENDQ;

typedef SVGENDQ *SVGENDQ_handle;
```

3.8 Proporcionalno integralni diferencionalni regulator

Modul PID regulatora je realizovan kao struktura. Deklaracija strukture, direktno preuzeta iz projektne datoteke pid_reg3.h data je u tekstu niže.

```

typedef struct
{
    _iq Ref;                                // Input: Reference input
    _iq Fdb;                                // Input: Feedback input
    _iq Err;                                // Variable: Error
    _iq Kp;                                 // Parameter: Proportional gain
    _iq Up;                                 // Variable: Proportional output
    _iq Ui;                                 // Variable: Integral output
    _iq Ud;                                 // Variable: Derivative output
    _iq OutPreSat;                          // Variable: Pre-saturated output
    _iq OutMax;                             // Parameter: Maximum output
    _iq OutMin;                             // Parameter: Minimum output
    _iq Out;                                // Output: PID output
    _iq SatErr;                            // Variable: Saturated difference
    _iq Ki;                                 // Parameter: Integral gain
    _iq Kc;                                 // Parameter: Integral correction gain
    _iq Kd;                                 // Parameter: Derivative gain
    _iq Up1;                               // History: Previous proportional output
    void (*calc)();                         // Pointer to calculation function
} PIDREG3;

typedef PIDREG3 *PIDREG3_handle;

```

3.9 Kontroler „ramp“ signala

Modul se koristi za pokretanje motora iz stanja mirovanja. Modul RAMP CTRL je realizovan kao struktura. Deklaracija strukture, direktno preuzeta iz projektne datoteke rmp_cntl.h data je u tekstu niže.

```
typedef struct
{
    _iq     TargetValue;           // Input: Target input (pu)
    Uint32 RampDelayMax;         // Parameter: Maximum delay rate (Q0)
    _iq     RampLowLimit;         // Parameter: Minimum limit (pu)
    _iq     RampHighLimit;        // Parameter: Maximum limit (pu)
    Uint32 RampDelayCount;       // Variable: Incremental delay (Q0)
    _iq     SetpointValue;        // Output: Target output (pu)
    Uint32 EqualFlag;            // Output: Flag output (Q0)
    void (*calc)();              // Pointer to calculation function
} RMPCNTL;

typedef RMPCNTL *RMPCNTL_handle;
```

3.10 Generator „ramp“ signala

Generiše „ramp“ signal u zavisnosti od kontrole signala. Realizovan je kao struktura. Deklaracija strukture, direktno preuzeta iz projektne datoteke rampgen.h data je u tekstu niže.

```
typedef struct
{
    _iq   Freq;                  // Input: Ramp frequency (pu)
    _iq   StepAngleMax;          // Parameter: Maximum step angle (pu)
    _iq   Angle;                 // Variable: Step angle (pu)
    _iq   Gain;                  // Input: Ramp gain (pu)
    _iq   Out;                   // Output: Ramp signal (pu)
    _iq   Offset;                // Input: Ramp offset (pu)

    void (*calc)();              // Pointer to calculation function
} RAMPGEN;

typedef RAMPGEN *RAMPGEN_handle;
```

3.11 Kalkulator faznog napona

Realizovan je kao struktura. Deklaracija strukture, direktno preuzeta iz projektne datoteke volt_calc.h data je u tekstu niže.

```

typedef struct
{
    _iq  DcBusVolt;           // Input: DC-bus voltage (pu)
    _iq  MfuncV1;            // Input: Modulation voltage phase A (pu)
    _iq  MfuncV2;            // Input: Modulation voltage phase B (pu)
    _iq  MfuncV3;            // Input: Modulation voltage phase C (pu)
    Uint32 OutOfPhase;       //Parameter:Out of Phase adjustment(0 or 1)
    _iq  VphaseA;            // Output: Phase voltage phase A (pu)
    _iq  VphaseB;            // Output: Phase voltage phase B (pu)
    _iq  VphaseC;            // Output: Phase voltage phase C (pu)
    _iq  Valpha;             // Output: Stationary d-axis phase voltage (pu)
    _iq  Vbeta;               // Output: Stationary q-axis phase voltage (pu)
    void (*calc)();          // Pointer to calculation function
} PHASEVOLTAGE;

typedef PHASEVOLTAGE *PHASEVOLTAGE_handle;

```

3.12 Estimator ugla rotora

Modul estimatora ugla je realizovan kao struktura. Deklaracija strukture, direktno preuzeta iz projektne datoteke smopos.h data je u tekstu niže.

```

typedef struct
{
    _iq  Valpha;              // Input: Stationary alfa-axis stator voltage
    _iq  Ealpha;                // Variable: Stationary alfa-axis back EMF
    _iq  Zalpha;                //Output: Stationary alfa-axis sliding control
    _iq  Gsmopos;              // Parameter: Motor dependent control gain
    _iq  EstIalpha;             //Variable:Estimated stationary alfa-axis stator
current
    _iq  Fsmopos;              // Parameter: Motor dependent plant matrix
    _iq  Vbeta;                  // Input: Stationary beta-axis stator voltage
    _iq  Ebeta;                  // Variable: Stationary beta-axis back EMF
    _iq  Zbeta;                  // Output: Stationary beta-axis sliding control
}

```

```

    _iq  EstIbeta;           // Variable: Estimated stationary beta-axis
stator current
    _iq  Ialpha;            // Input: Stationary alfa-axis stator current
    _iq  IalphaError;       // Variable: Stationary alfa-axis current error
    _iq  Kslide;           // Parameter: Sliding control gain
    _iq  Ibeta;            // Input: Stationary beta-axis stator current
    _iq  IbetaError;       // Variable: Stationary beta-axis current error
    _iq  Kslf;              // Parameter: Sliding control filter gain
    _iq  Theta;             // Output: Compensated rotor angle
    void (*calc)();         // Pointer to calculation function
} SMOPOS;

typedef SMOPOS *SMOPOS_handle;

```

3.13 Estimator brzine rotora

Modul estimtora brzine rotora je realizovan kao struktura: Deklaracija strukture, direktno preuzeta iz projektne datoteke speed_est.h data je u tekstu niže.

```

typedef struct
{
    _iq EstimatedTheta;      // Input: Electrical angle (pu)
    _iq OldEstimatedTheta;   // History: Electrical angle at previous step (pu)
    _iq EstimatedSpeed;      // Output: Estimated speed in per-unit (pu)
    Uint32 BaseRpm;          // Parameter: Base speed in rpm(Q0)
    _iq21 K1;                // Parameter: Constant for differentiator(Q21)
    _iq K2;                  // Parameter: Constant for low-pass filter (pu)
    _iq K3;                  // Parameter: Constant for low-pass filter (pu)
    int32 EstimatedSpeedRpm; // Output: Estimated speed in rpm (Q0)
    void (*calc)();           // Pointer to the calculation function
} SPEED_ESTIMATION;          // Data type created

typedef SPEED_ESTIMATION *SPEED_ESTIMATION_handle;

```

4. Programsko rešenje

4.1 Programska podrška

4.1.1 2xPM_Motors modul

Glavni modul programske podrške Texas Instruments-ovog rešenja je realizovan kao automat sa jednom prekidnom rutinom, MainISR(), koja izvršava vekorsku kontrolu (potrebne transformacije, regulacije, konverzije, procene, i upravljanje imulsno širinskim modulatorom). Glavna prekidna rutina se pokreće jednom u svakom PWM ciklusu. Slika 3.1 predstavlja blok šemu prekidne rutine. Na TI rešenje dodata je podrška za serijsku komunikaciju, kao i automatski prelaz iz otvorene petlje (“*ramp up*” moda) u zatvorenu petlju (vektorska kontrola). Kada motor postigne dovoljnu brzinu (dovoljno povratne elektromotorne sile), moguće je odrediti brzinu motora. Eksperimentalno je odredjeno na kojim brzinama je optimalan prelaz izmedju otvorene i zatvorene petlje, i implementiran je automatski prelaz izmedju ova dva načina upravljanja motorom.

```
if(lsw1 != 0) { //lsw1==0 motor stoji
    if(speed3.EstimatedSpeedRpm > 700) {
        lsw1=2; //lsw1==1 otvorena petlja
    }

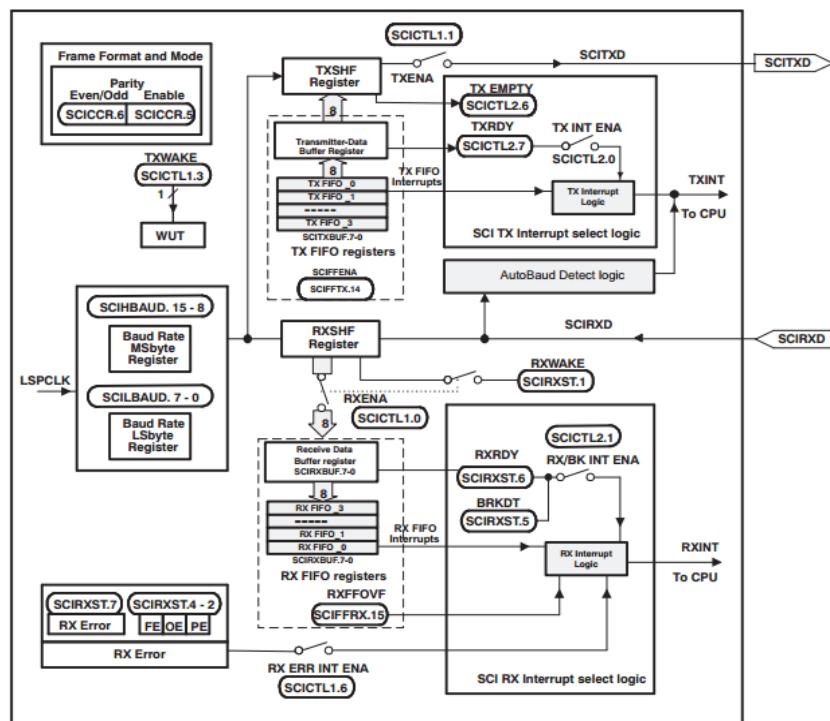
    if(speed3.EstimatedSpeedRpm < 500) {
        lsw1=1; //lsw1== zatvrena petlja
    }
}
```

4.1.2 Asinhrona seriska komunikacija

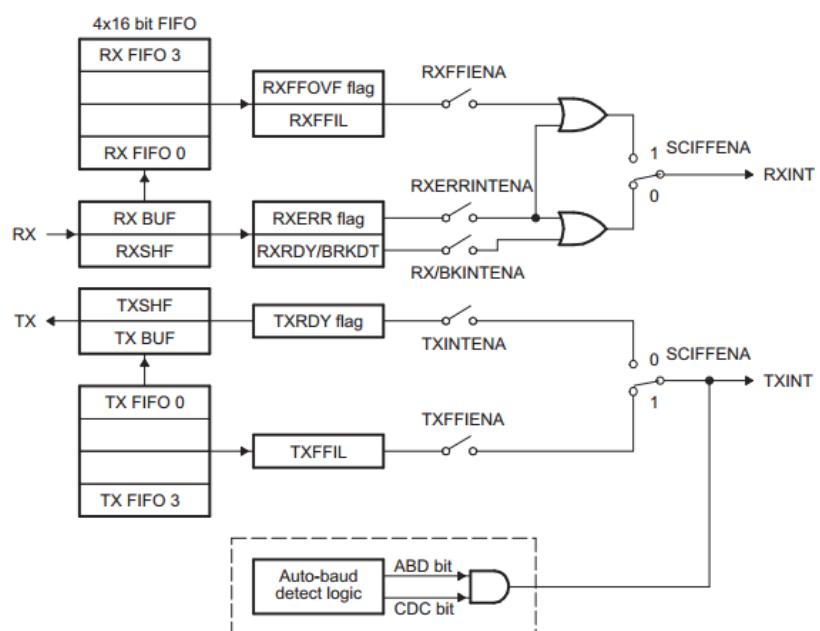
Univerzalna asinhroni prijemnik predajnik je dvožični asinhrona serijska komunikaciona sprega, poznata kao UART sprega, ili SCI sprega. UART modul pruža podršku za komunikaciju izmedju CPU i drugih asinhronih periferija koje koriste standardni NRZ (non-return-to-zero) format. [15]

Odlike UART sprege na Piccolo F28035 razvojnoj ploči:

- Dva spoljna konektora:
 - SCITXD za slanje,
 - SCIRXD za primanje,
 - mogu se koristiti kao opštenamenski ulazi/izlaz ako se ne koriste za serijsku komunikaciju
- Baud rate programabilan do 64 hiljade različitih komunikacionih brzina
- Format reči:
 - Jedan startni bit
 - Duzina reči programaski podešiva od jednog do osam bita
 - Opcioni bit parnosti (parno,neparno,bez)
- Detekcija greške parnosti
- Polu dupleks(*half-duplex*) ili dupleks(*full-duplex*)
- Operacije slanja i primanja mogu se obaviti putem prekida, ili preko ispitivanja (pooling)
- NRZ format(bez povratka na nulu)



Slika 4.1 Blok dijagram modula za serijsku komunikaciju



Slika 4.2 Dijagram prekida serijske komunikacije

Serijska komunikacija je realizovana u UartSci modulu. Modul je pisan u C programskom jeziku. Sastoje se sledećih funkcija:

Naziv funkcije	Opis
void sendMessage()	Funkcija priprema poruku za slanje preko serijske magistrale.
void UartSciControl	Čita podatke iz kružnog bafera, i u zavisnosti od primljene poruke, izvršava odgovarajući komandu. Prima podatke do terminacijskog znaka. Po prijemu neispravne komande, prazni kružni bafer, i počinje prijem nove komande.
void UartSciaInit()	Postavlja odgovarajuće vrednosti u registar modula za serijsku komunikaciju, koji su neophodni za ispravnu komunikaciju.
__interrupt void sciaTxFifoIsr(void)	Prekidna rutina za slanje podataka, ispisuje iz kružnog bafera, i prazni ga.
__interrupt void sciaRxFifoIsr(void)	Prekidna rutina za prijem podataka, koja isčitava podatke iz prijemnog registra, i upisuje ih u kružni bafer, koji koristi UartSciControl funkcija.

Tabela 4.1 Funkcije programske podrške

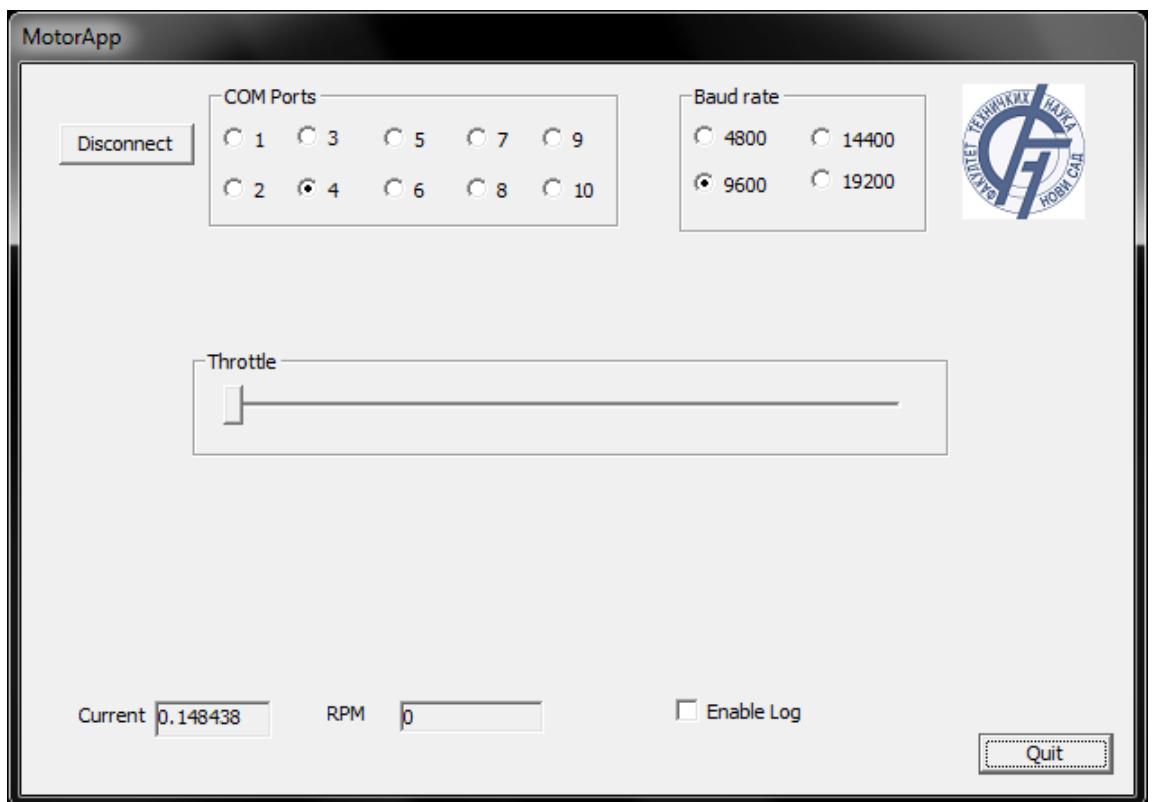
Programska podrška prima znakove u kruzni bafer do prijema terminacionog znaka (;). Po prijemu terminacionog znaka, vrši se provera da li je format primljene poruke ispravan.

Komanda	Opis
speed:0.0;	Podešava brzinu u opsegu od 0,0 do 0,9.
adc;	Zahteva od programske podrške da pošalje podatak o vektoru struje. Vektor struje se salje kao 8bitna vrednost, u formatu sa fiksnim zarezom.
rpm;	Zahteva od programske podrške da pošalje podatak procenjenom broju obrtaja motora. Šalje se kao dve 8-bitne vrednosti, koje predstavljaju gornji i donji deo 16-bitne celobrojne promenjive.

Tabela 4.2 Spisak komandi

4.2 Grafička korisnička sprega

Grafička korisnička sprega omogućava korisniku povezivanje sa razvojnom pločom, podešavanje serijske komunikacije (podesavanje prolaza, i brzine komunikacije), i kontrolu motora putem klizača, kao i prikaz amplitude vektora struje i broja obrtaja. Takodje omogućava zapis stanja sistema u datoteku za evidenciju(*log file*).



Slika 4.3 Izgled grafičke korisničke sprege

Grafička korisnička sprega realizovana u C++ programskom jeziku, koristeci MFC (Microsoft Foundation Class) biblioteku, koja omotava delove Windows-ove aplikacione programske sprege u C++ klase.

Takodje, korišten je i omotač seriskog porta, (serialport wrapper), koji uprošćava korišćenje Windows-ovih komadni za rad sa serijskim prolazom. Razvijen je od strane *PJ Naugther*. [18]

Grafička korisnička sprega se sastoji od jednog dijalog-prozora, i automatski generisane CMotorAppDlg klase.

U klasu su dodate kontrolne promenjive, za čitanje vrednosti radio dugmadi kojima se podešavaju parametri serijske komunikacije, kao i kontrolne promenjive preko kojih se ispisuje vrednosti vektora struje i broja obrtaja motora. Dodata je jedna instanca klase CSerialPort, preko koje se vrši povezivanje i komunikacija putem serijskog porta.

Pritiskom na dugme "Connect", otvara se izabrani prolaz, i aplikacija se povezuje sa programskom podrskom razvojne ploče. Pomeranjem klizača šalju se komande preko UART sprege, i kontroliše se brzina motora.

Podatke o vektoru struje, i procenjenoj brzini motora se dobavljuju na pola sekunde. Pritiskom na dugme za povezivanje, pokreće se tajmer, koji pri isteku zadatog interval od 500mS, poziva funkciju koja putem UART komandi pribavlja podatke o procenjenoj brzini, i vektoru struje.

Naziv metode	Opis
int getPortFromRadioBox()	Koristi se za čitanje radio dugmadi, za dobijanje podatka o izabranom portu, preko kog se vrši komunikacija.
DWORD getBaudRateFromRadioBox()	Za čitanje zeljene brzine komunikacije iz radio dugmića.
afx_msg void OnTimer(UINT_PTR nIDEvent);	Tajmerska rutina, koja se izvršava na svakih pola sekunde, služi za dobavljanje informacija o vektoru struje i procenjenoj brzini obrtanja motora.
afx_msg void OnBnClickedConnect();	Povezuje se sa programskom podrškom, pokreće tajmere, menja labelu dugmeta u "Disconnect", i omogucava korisniku da pomera klizač za upravljanje brzinom motora. Pritiskom na dugme kad je labela "Disconnect", zaustavlja se tajmer, zadata brzina se postavlja na 0, prekida se veza, I menja se labela dugmeta u "Connect".
void StopTimers();	Funkcija za zaustavljanje tajmera.
void StartTimers();	Funkcija za pokretanje tajmera.
afx_msg void OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar);	Funkcija koja reaguje na pomeraj klizača za podešavanje brzine, salje komadnu preko UART sprege za postavljanje zeljene brzine.

Tabela 4.3 Funkcije grafičke korisničke sprege

5. Ispitivanje i verifikacija



Slika 5.1 Razvojna ploča Multi-Axis DMC

Rad je radjen na Texas Instruments-ovoj **Multi-Axis Digital Motor Control Kit** razvojnoj platformi. Komplet sadrži:

- F28035 Piccolo kontrolnu karticu
- Multu-Axis DMC matičnu ploču.
- Adapter na 24V
- 2 motora jednosmerne struje bez četkica, **Anaheim Automation BLY172S-24V-4000**

Osobine Multi-Axis Digital Motor Control razvojne ploče:

- Vektorska kontrola 2 motrora bez senzora koristeći Texas Instruments-ov DRV8402 IPM modul

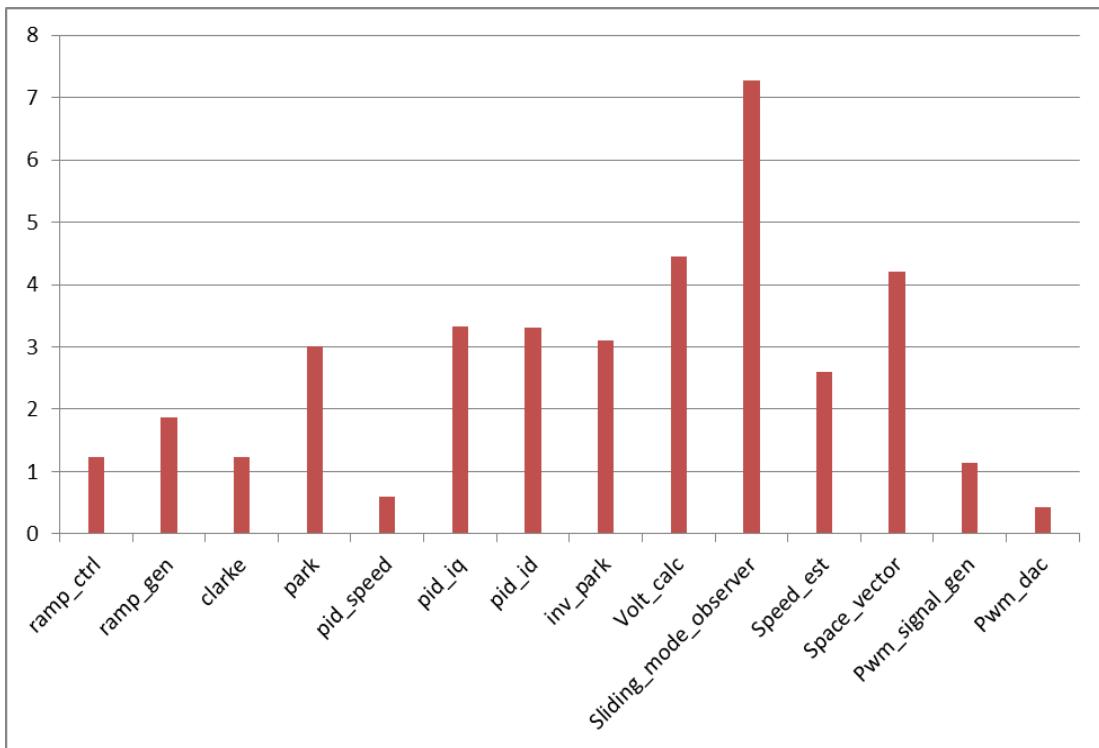
- o Sinusna kontrola 2 motora sa stalnim magnetima bez četkica
- o 10KHz frekvencija prekidanja po invertorskoj fazi
- o Približno 90% efikasnost
 - Digitalna kontrola u zatvorenoj petlji sa povratnom spiegom koristeći C2000's integrisano kolo, i periferije: impulsno širinski modulator i analogno digitalni konvertor
 - JTAG emulacija
 - Naponska i strujna zaštita
 - UART komunikacija
 - Frekvencija PWM: 10 kHz

5.1 Ispitivanje i analiza programske podrške

Analiza zauzetosti procesora pri izvršavanju blokova programske podrške za vektorskiju kontrolu izvršena je na tri načina. Brojanjem ciklusa procesora iz Code Composer Studio-a, tajmerom koji broji otkucaje sistemskog takta pobude, i podizanjem/spuštanje pina opšte namene pri ulazu/izlazu is prekidne servisne rutine. Osciloskopom je izmereno trajanje impulsa na opštenamenskom pinu, i može se primetiti da vrednosti dobijene u sva tri slučaja veoma slične. U Tabeli 3.1 su date srednje vrednosti vremena izvršenja blokova programske podrške za vektorskiju kontrolu motora. Jedan PWM ciklus traje 100 mikrosekundi. Neophodno je osigurati da se sva dodatna obrada izvrši pre novog PWM ciklusa, inače može doći do nepouzdanog rada, ako processor nije u stanju da na vreme završi posao iz glavne prekidne rutine.

Naziv modula	Broj CPU ciklusa	Vreme [μ S]
ramp_ctrl	74	1.233
ramp_gen	112	1.867
clarke	74	1.233
park	181	3.017
pid_speed	36	0.6
pid_iq	200	3.334
pid_id	199	3.317
inv_park	186	3.1
Volt_calc	267	4.45
Sliding_mode_observer	436	7.268
Speed_est	156	2.6
Space_vector	252	4.2
Pwm_signal_gen	79	1.131
Pwm_dac	23	0.433
Ukupno	2671	44.525

Tabela 5.1 Analiza zauzetosti procesora pri izvršavanju blokova programske podrške za vektorskiju kontrolu



Slika 5.2 Grafički prikaz zauzetosti procesora u μ s

Za početno ispitivanje UART sprege programske podrške je korišćen programski alat “Terminal v1.9 – 20041226 – by Br@y++“.

Slanjem odgovarajućih komandi preko alata “Terminal” utvrđeno je da se motor ponaša u skladu sa zadatim komandama (podešavanje brzine, zaustavljanje i pokretanje motora).

5.2 Ispitivanje i verifikacija grafičke korisničke sprege

Funkcionalna verifikacija grafičke korisničke sprege koja upravlja motorom je uradjena u više eksperimenata.

Pre konekcije proverava se da li je izabrani prolaz otvoren. Ako prolaz nije otvoren, korisnik se preko dijaloga obaveštava da je izabrani port nedostupan.

Uspesna komunikacija zahteva da i programska podrška i grafička korisnička sprege budu usaglašene oko parametara komunikacije, tj podešene na identičan način.

Podešavanja:

- Brzina prenosa (*baud rate*): - 9600
- Bitova podataka: 8
- Parnost: ne koristi se bit parnosti

- Stop bita: 1
- Rukovanje(handshaking) : bez rukovanja

Ukoliko podešavanja nisu ista sa obe strane komunikacije, može doći do nepredvidjenog rada i izuzetaka.

Definisan je jednostavan protocol za komunikaciju, grafička korisnička sprega šalje komandu za promenu brzine kada korisnik promeni poziciju klizača, a grafička korisnička sprega na svakih pola sekunde palje zahtev za podatcima o stanju sistema.

Procena brzine obrtaja motora zavisi od procene ugla rotora, pa ne radi dok se motor ne zavrти dovoljnom brzinom, (da generise dovoljno povratne elektro motorne sile), tako da na niskim brzinama procena brzine obrtanja motora ne funkcioniše na ispravan način. Na višim brzinama procena ispravno radi.

Eksperimentalno je utvrđeno da programska podrška prima komande, i odgovarajuće reaguje na zadate komande.

Verifikovano je da sistem reaguje na komande, prikazuje vrednosti vektora struje i procenjen broj obrtaja u minuti, i na zahtev korisnika upisuje podatke u datoteku eksperimenta(*log file*)

6. Zaključak

U radu je ispitano TI rešenje vektorske kontrole motora sa stalnim magnetima bez četkica. Dodata je podrška za serijsku komunikaciju i kontrolu motora putem UART sprege. Omogućen je automatski prelaz izmedju otvorene i zatvorene petlje upravljanja. Napravljena je grafička korisnička sprega koja korisniku omogućava kontrolu motora sa strane personalnog računara, i prikazuje staje sistema(vektor struje i broj obrtaja motora u minuti). Korisnik takođe ima mogućnost uključivanja zapisa stanja sistema u datoteku evidencije(*log file*).

Motori jednosmerne struje sa stalnim magnetima bez četkica kojima se vektorski upravlja se sve više koriste umesto motora jednosmernih struja sa skalarnim upravljanjem. Današnji trendovi razvoja su takvi da u savremenim pogonima motori jednosmerne struje sa stalnim magnetima bez četkica polako potiskuju iz upotrebe asinhronih motora. [9] Motori sa stalnim magnetima imaju bolje performanse, ali zbog stalnih magneta su skuplji od asinhronih motora. Asinhroni motori su i dalje u širokoj upotrebi zbog svoje jednostavnosti, i niže cene u odnosu na motore sa stalnim magnetima.

Algoritmi vektorske kontrole motora se i dalje razvijaju i unapređuju, povećava se robusnost sistema, minimiziraju gubitci, smanjuje cena, itd. [10] Sa daljim napretkom i razvojem, i padom cena digitalne elektronike i mikroprocesora, očekuje se da će vektorska kontrola postati primarna metoda kontrole motora. [11]

Utvrđeno je da TI integrisano kolo Piccolo F28035 ima dovoljno procesorskih resursa za vektorskiju kontrolu, sa mogućnošću da se obave dodatni proračuni bez negativnog uticaja na vektorsko upravljanje.

Dalji smer istraživanja se može razvijati u pravcu implementacije trapezoidne kontrole, koja daje niže performasne u odnosu na vektorskiju kontrolu, ali zahteva manje akvizicionih i procesorskih resursa. U tom smislu analizirano integrisano kolo bi moglo uspešno da poluži kao

osvna platforma implementacije sistema čiji bi konačni cilj bio jeftino i pouzdano namensko rešenje za trapezno upravljanje jednosmernim motorima bez četkica, bez senzora pozicije.

7. Literatura

- [1] Masao Yano: *History of Power Electronics for Motor Drives in Japan*, Retrieved 18 April 2012
- [2] K. Hasse, *Zum Dynamischen Verhalten der Asynchronmaschine bei Betrieb mit variable Standerfrequenz und Standerspannung*, ETZ-A, Bd.89, H.4, pp.77-81, 1968.
- [3] F. Blaschke, *Das Prizip der Feldorientierung , Die Grundlage fur die TRNSVEKTOR-Regelung von Asynchronmaschinen*, Siemens Zeitschrift, 45, p.757, 1971
- [4] Petar Matić, Branko Blanuša, Slobodan N. Vukosavić, *Direktna kontrola momenta i vektorsko upravljanje u mikroprocesorskom upravljanju elektromotornim pogonima*, INFOTEH-JAHORINA, Vol. 2, Ref. D-2, p. 227-231, March 2002.
- [5] Slobodan N. Vukosavić, *Električne mašine*, Univerzitet u Beogradu, Elektrotehnički fakultet, Akademska misao, Beograd 2010
- [6] Slobodan N. Vukosavić, *Digitalno upravljanje električnim pogonima*, Akademska misao, Beograd,2003
- [7] W. C. Duesterhoeft, Max W. Schulz and Edith Clarke (July 1951). "Determination of Instantaneous Currents and Voltages by Means of Alpha, Beta, and Zero Components". Transactions of the American Institute of Electrical Engineers 70 (2): 1248–1255. doi:10.1109/T-AIEE.1951.5060554. ISSN 0096-3860
- [8] R.H. Park, *Two Reaction Theory of Synchronous Machines*, AIEE Transactions 48:716-730 (1929).
- [9] H. Holtz, "Pulsewidth Modulation for Electronic Power Conversion", Proceedings of the IEEE, Vol. 82, No. 8, Aug. 1994, pp. 1194-1214
- [10] Slobodan N. Vukosavić, *Projektovanje adaptivnog mikroprocesorskog upravljanja brzinom i pozicijom asinhronog motora*, doktorska disertacija, Univerzitet u Beogradu, 1989

- [11] Branko D. Blanuša, *Algoritam za minimizaciju snage gubitaka vektorski regulisanog asinhronog pogona zasnovan na primjeni fazi logike*, magistarski rad, Univerzitet u Banjaluci, 2002.
- [12] Bose, Bimal K, "The Past, Present, and Future of Power Electronics", Industrial Electronics Magazine, IEEE 3 (2): 11. (June 2009).
- [13] Holtz, J, "Sensorless control of induction motor drives". Proceedings of the IEEE 90 (8): 1359–1394. doi:10.1109/jproc.2002.800726, (Aug 2002).
- [14] Sensorless Field Oriented Control of Multiple Permanent Magnet Motors, Application Report ,SPRABQ9, Dallas, Texas, July 2013,
<http://www.ti.com/lit/an/sprabq9/sprabq9.pdf>
- [15] TMS320F2803x Piccolo System Control and Interrupts, Reference Guide, Literature Number: SPRUGL8C, ,Dallas, Texas, May 2009–Revised February 2013
<http://www.ti.com/lit/ug/sprugl8c/sprugl8c.pdf>
- [16] TMS320x2802x, 2803x Piccolo Analog-to-Digital Converter (ADC) and Comparator, Reference Guide, Literature Number: SPRUGE5F, Dallas, Texas, December 2008– Revised December 2011
<http://www.ti.com/lit/ug/spruge5f/spruge5f.pdf>
- [17] TMS320x2802x, 2803x Piccolo Enhanced Pulse Width Modulator (ePWM) Module, Reference Guide, Literature Number: SPRUGE9E, Dallas, Texas, December 2008– Revised March 2011
<http://www.ti.com/lit/ug/spruge9e/spruge9e.pdf>
- [18] U radu je koršten CSerialPort v1.03 Serial Port Wraper, preuzet sa: <http://www.codeproject.com/Articles/382/CSerialPort-v-Serial-Port-Wrapper>
- [19] Miloš Nikolić, *Jedno rešenje fizičke arhitekture za upravljanje jednosmernim motorom bez četkica sa četiri elektronska prekidača*, Magistarski rad, Novi Sad, 2014
- [20] Slike transformacija su preuzete sa:
Microsemi's Park, Inverse Park and Clarke, Inverse Clarke Transformations MSS Software Implementation User Guide
http://www.microsemi.com/index.php?option=com_docman&task=doc_download&gid=132799
- [21] M.P. Kazmierkowski, R. Krishnan, and F. Blaabjerg *Control in Power Electronics: Selected Problems*. San Diego: Academic Press. (2002). ISBN 978-0-12-402772-5